Abstract—Commonsense knowledge (CSK) about concepts and their properties is helpful for AI applications. Prior works, such as ConceptNet, have compiled large CSK collections. However, they are restricted in their expressiveness to subject-predicate-object (SPO) triples with simple concepts for S and strings for P and O. This paper presents a method called ASCENT++ to automatically build a large-scale knowledge base (KB) of CSK assertions, with refined expressiveness and both better precision and recall than prior works. ASCENT++ goes beyond SPO triples by capturing composite concepts with subgroups and aspects, and by refining assertions with semantic facets. The latter is essential to express the temporal and spatial validity of assertions and further qualifiers. Furthermore, ASCENT++ combines open information extraction (OpenIE) with judicious cleaning and ranking by typicality and saliency scores. For high coverage, our method taps into the large-scale crawl C4 with broad web contents. The evaluation with human judgments shows the superior quality of the ASCENT++ KB, and an extrinsic evaluation for QA-support tasks underlines the benefits of ASCENT++. A web interface, data, and code can be accessed at https://ascentpp.mpi-inf.mpg.de/.

Index Terms—Commonsense knowledge, knowledge base construction

1 INTRODUCTION

Motivation. Commonsense knowledge (CSK) is a longstanding goal of AI [1], [2], [3], [4]: equip machines with structured knowledge about everyday concepts and their properties (e.g., elephants are big and eat plants, buses carry passengers and drive on roads) and about typical human behavior and emotions (e.g., children love visiting zoos, children enter buses to go to school). In recent years, research on the automatic acquisition of CSK assertions has significantly been advanced, and several commonsense knowledge bases (CSKBs) of considerable size have been constructed (see, e.g., [5], [6], [7], [8]). Use cases for CSK include particularly language-centric tasks such as question answering, conversational systems, and text generation (see, e.g., [9], [10], [11], [12], [13]).

Examples: Question-answering systems often need CSK as background knowledge for robust answers. For example, when a child asks “Which zoos have habitats for T-Rex dinosaurs?” , the system should point out that 1) dinosaurs are extinct, and 2) they can be seen in museums, not in zoos. Dialogue systems should not only generate plausible utterances from a language model but should also be situative, understand metaphors and implicit contexts and avoid blunders. For example, when a user says “tigers will soon join the dinosaurs” , the machine should understand that this refers to an endangered species rather than alive tigers invading museums.

This paper aims to advance the automatic acquisition of CSK assertions from online content to bring better expressiveness, higher precision, and broader coverage.

State-of-the-Art and its Limitations. Large KBs like DBpedia, Wikidata, or Yago focus on encyclopedic knowledge of individual entities like people or places and are sparse on general concepts [14]. Notable projects that concentrate on CSK include ConceptNet [5], WebChild [6], Mosaic TupleKB [7], Quasimodo [8], ATOMIC [15], TransOMCS [16] and AutoTOMIC [17]. They are all based on subject-predicate-object (SPO) triples as knowledge representation and have significant shortcomings:

- Expressiveness for subject (S): As subjects, prior CSKBs strongly focus on simple concepts expressed by single nouns (e.g., “bus”, “elephant”, “car”, “trunk”). This misses semantic refinements (e.g., “diesel bus” versus “electric bus”) that lead to different properties (e.g., “polluting” versus “green”) and is also prone to word-sense disambiguation problems (e.g., “elephant trunk” versus “car trunk”). Even when CSK acquisition considers multi-word phrases, it still lacks the awareness of semantic relations among concepts. Hypernymy lexicons like WordNet or Wiktionary are also very sparse on multi-word concepts. With these limitations, word-sense disambiguation does not work robustly; prior attempts showed mixed results at best (e.g., WebChild [6], TupleKB [7]).

- Expressiveness for predicate (P) and object (O): Predicates and objects are treated as monolithic strings, such as:
  - A1: {bus, is used for, transporting people};
  - A2: {bus, is used for, bringing children to school};
  - A3: {bus, carries, passengers};
  - A4: {bus, drops, visitors at the zoo on the weekend}.
This misses the equivalence of assertions A1 and A3 and cannot capture the semantic relation between A1 and A2, namely, A2 refining A1. Finally, the spatial facets of A2 and A4 are cluttered into unrelated strings, and the temporal facet in A4 is not explicit either. The alternative of restricting predicate (P) to a small number of pre-specified relations (e.g., ConceptNet [5], TupleKB [7]) and object (O) to concise phrases comes at the cost of much lower coverage.

- Quality of CSK assertions: Some of the major CSKBs have prioritized precision (i.e., the validity of the assertions) but have fairly limited coverage (e.g., ConceptNet [5], TupleKB [7]). Others have broader coverage but include many noisy if not implausible assertions (e.g., WebChild [6], Quasimodo [8]). Very few have paid attention to the saliency of assertions, i.e., the degree to which statements are common knowledge, as opposed to merely capturing many assertions. However, projects along these lines (e.g., ConceptNet [5], ATOMIC [15]) fall short in coverage.

Ascent++ aims to overcome these limitations of prior works while retaining their positive characteristics. In particular, we aim to reconcile high precision with wide coverage and saliency. Like TupleKB [7] and Quasimodo [8], we desire to acquire open assertions (as opposed to pre-specified predicates only) but strive for more expressive representations by refining subjects and capturing semantic facets of assertions.

Furthermore, we also provide a canonicalized version of the resulting CSKB in the ConceptNet schema, thus enabling direct use in applications relying on that fixed schema.

Approach. We present the Ascent++ method for acquiring CSK assertions with refined semantics from web contents. Ascent++ operates in two phases: (i) scalable extraction from a large web corpus and (ii) aggregation and consolidation. In the first phase, Ascent++ processes the C4 crawl [18], a collection of 365 million English web pages. Ascent++ extracts OpenIE-style tuples by carefully designed dependency-parse-based rules, taking into account assertions for subgroups and aspects of target subjects. The extractor uses cues from prepositional phrases and adverbs to detect semantic facets and uses supervised classification for eight facet types.

In the second phase, on a per-subject basis, Ascent++ identifies relevant web pages based on embedding similarity to reference Wikipedia articles, this way being able to distinguish homonyms like “bus (public transport)” versus “bus (network topology)”. Assertions are iteratively grouped and organized using embedding-based similarity. OpenIE-style assertions are canonicalized into the established ConceptNet schema. Finally, a supervised machine learning model ranks the resulting statements by saliency and typicality scores.

We ran Ascent++ on the C4 crawl for 10,000 salient concepts from ConceptNet as target subjects. To evaluate the intrinsic quality of the resulting CSKB, we obtained human judgments for a large sample. Our CSKB significantly improves over automatically-built state-of-the-art CSK collections in terms of precision and relative recall.

In addition, we performed an extrinsic evaluation in which commonsense knowledge was used to support language models in question answering. Using three different settings and six different CSKBs, Ascent++ significantly outperformed language models without this CSK background knowledge in 2 of the 3 settings, and was best or second best among all 6 CSKBs in all three cases.

Contributions. This work’s key contributions are:

1. An expressive model for commonsense knowledge with advanced semantics, subgroups of subjects and faceted assertions as first-class citizens, and scores for typicality and saliency (see Section 3).
2. An unsupervised automated method for populating the model with high-quality CSK assertions by large-scale web content extraction and various techniques for aggregation and cleaning (see Section 4).
3. Constructing and publicly releasing a high-quality CSKB with 2 million assertions for 10,000 important concepts (see Section 5).

A web interface to the Ascent++ KB, with downloadable data and code, is accessible at https://ascentpp.mpi-inf.mpg.de/

Prior Publication. This paper substantially extends an earlier conference paper [19]. Major extensions are:

1. The extraction method is completely re-worked. The earlier version operated solely on top-ranked query results from search engine APIs, whereas Ascent++ processes a massive web crawl. This involves new techniques for scalability and quality control in the presence of very noisy web contents (see Sections 4.2, 4.3, and 4.6).
2. We extended the data model to include a new dimension of typicality scores, computed by a supervised regression model that considers assertion facets (see Section 4.7).
3. We improved our techniques for canonicalizing assertions, added automatic mapping onto the schema of ConceptNet, and updated the evaluation. This includes a new module for statement canonicalization, enhanced clustering by leveraging Sentence-Transformers [20], and evaluation with GPT-3 [21] (see Sections 4.4, 4.5, and 6.2).
4. Due to this enhanced methodology, the Ascent++ KB improves over the previous Ascent KB [19] in KB scope and quality. Ascent++ has 25% higher typicality, and 14.6% higher saliency (for top-10 assertions per subject) and 36.6% higher relative recall (see Section 6.1.3).

2 Related Work

Commonsense Knowledge Bases (CSKBs). CSK acquisition has a long tradition in AI (e.g., [3], [22], [23], [24]).

The seminal Cyc project was the first to construct a large KB that captures commonsense knowledge, based on hand-crafting assertions by a team of knowledge engineers [3], [9]. OpenCyc [4] was released as a non-commercial version of Cyc, with much smaller size, though. The last version of this KB, OpenCyc 4.0, was released in 2012. However, this project has been discontinued, and no research license is available anymore. Therefore, we could not include Cyc or OpenCyc in our study.
More recently, a number of projects have constructed large-scale collections that are publicly available, such as ConceptNet [5], WebChild [6], TupleKB [7], Quasimodo [8], ATOMIC [15], [25], TransOMCS [16] and AutoTOMIC [17]. ConceptNet [5] combined CSK collected by human crowdsourcing and knowledge from existing resources such as Cyc, OpenCyc, WordNet, and Wiktionary. This CSKB contains highly salient information for a few pre-specified predicates (isa/type, part-whole, used for, capable of, location of, plus lexical relations such as synonymy, etymology, derived terms, etc.), and it is arguably the most widely used CSKB. However, it has limited coverage on many concepts and its ranking of assertions, based on the number of crowdsourcing inputs, is very sparse and unable to discriminate salient properties against atypical or exotic ones (e.g., listing “tree”, “garden”, and “the bible” as locations of “snake”, with similar scores). ConceptNet does not properly disambiguate concepts, leading to incorrect assertion chains like (elephant, HasPart, trunk); (trunk, LocationOf, spare tire).

WebChild [6], TupleKB [7], and Quasimodo [8] devised fully automated methods for CSKB construction. They use judiciously selected text corpora (incl. book n-grams, image tags, and QA forums) to extract large amounts of SPO triples. WebChild builds on hand-crafted extraction patterns, and TupleKB and Quasimodo rely on open information extraction (OpenIE) with subsequent cleaning. All three are limited to SPO triples.

ATOMIC [15], [25] and ASER [26], [27] are recent projects on event-centered CSKB construction. While ATOMIC is entirely based on a large-scale human compilation, assertions in ASER were extracted automatically from large text corpora using dependency patterns. Both resources use fixed relations (such as xNeed, xWant, isAfter, or isBefore in ATOMIC, and Reason, Result, Precedence, or Succession in ASER) and are also limited to triples. More recently, TransOMCS [16] mined concept-centric triples from ASER. TransOMCS used ConceptNet assertions as seed samples for pattern extraction, hence having the same pre-specified predicates as ConceptNet.

Other notable CSKB projects include CSKG [28] and GenericsKB [29]. CSKG is an attempt to combine seven different CSK resources into an integrated knowledge graph. GenericsKB, on the other hand, drops attempts at structuring assertions and instead focuses on collecting generic sentences on a per-subject basis.

Our preliminary paper on this work, ASPECT [19], [30], developed an open information extraction methodology that captures semantic facets and multi-word compounds as subjects, enabling finer-grained knowledge representation and avoiding common disambiguation errors. For knowledge source discovery, ASPECT used commercial search engine APIs. This choice ensured precision but reduced recall and strictly limited the scalability on monetary grounds. The present ASPECT++ approach overcomes this limitation by using a huge web crawl instead, with several orders of magnitude larger input.

**Taxonomy and Meronymy Induction.** The organization of concepts in terms of subclass and part-whole relationships, termed hypernymy and meronymy, has received great attention in NLP and web mining (e.g., [31], [32], [33], [34], [35], [36], [37], [38]). The hand-crafted WordNet lexicon [39] organizes over 100k synonym sets for these relationships, although meronymy is sparsely populated.

Recent methods for large-scale taxonomy induction from web sources include WebsIsADB [38], [40] building on Hearst patterns and other techniques, and the industrial GIANT ontology [41] based on neural learning from user-action logs and other sources.

Meronymy induction at a large scale has been addressed by [29], [42], [43] with pre-specified and automatically learned patterns for refined relations like physical-part-of, member-of, and substance-of.

Our approach includes relations of both kinds by extracting knowledge about salient subgroups and aspects of subjects. In contrast to typical taxonomies and part-whole collections, our subgroups include many multi-word phrases: composite noun phrases (e.g., “forest elephant”, “elephant keeper”) and adjectival and verbal phrases (e.g., “male elephant”, “working elephant”). Aspects cover additional refinements of subjects that do not fall under taxonomy or meronymy (e.g., “elephant’s diet”, “elephant habitat”).

Expressive Knowledge Representation and Extraction.** Modalities such as “always”, “often”, “rarely”, and “never” have a long tradition in AI research (e.g., [44]), based on various kinds of modal logics or semantic frame representations, and semantic web formalisms can capture context using, e.g., RDF* or reification [45]. While such expressive knowledge representations have been around for decades, there has hardly been any work that populated KBs with such refined models, notable exceptions being the Knext project [46] at a small scale and OntoSenticNet [47] with a focus on affective valence annotations.

Other projects have pursued different kinds of contextualizations for CSK extraction, notably [48], which scored natural language sentences on an ordinal scale covering the spectrum “very likely”, “likely”, “plausible”, “technically possible”, and “impossible”, [49] with probabilistic scores, and the Dice project [50] which ranked assertions along the dimensions of plausibility, typicality, markedness, and saliency.

Semantic role labeling (SRL) is a representation and methodology where sentences are mapped onto frames (often for certain types of events), and respective slots (e.g., agent, participant, instrument) are filled with values extracted from the input text [51], [52], [53]. Recently, this paradigm has been extended towards facet-based open information extraction, where extracted tuples are qualified with semantic facets like location and mode [54], [55]. ASPECT and ASPECT++ have built on this general approach but extended it in various ways geared for the case of CSK: focusing on specifically relevant facets, refining subjects by subgroups and aspects, and aiming to reconcile precision and coverage for concepts as target subjects.

**Pre-Trained Language Models and CSK.** Pre-trained language models (LMs) like BERT [56] and GPT [21], [57] have revolutionized NLP with remarkable advances on many tasks, including those related to commonsense knowledge. For instance, the LAMA probe [58] aimed to extract commonsense knowledge from masked language models via a cloze-style QA task. COMET [59] is an autoregressive language model fine-tuned on existing CSK resources (e.g., ConceptNet and ATOMIC) that is used to predict objects
for given subject-predicate pairs, and there are several related works [60], [61]. However, the quality of COMET’s generated CSK assertions is often significantly lower than that of its training resources, either manually-curated or web-extracted ones [62].

More recently, [17] introduced a knowledge distillation method that extracts CSK from a general language model, GPT-3 [21], and uses the extracted knowledge to train a smaller commonsense model, COMET [59], which was shown to perform better than GPT-3 in terms of commonsense capabilities. The CSK resource extracted from GPT-3, called AutoTOMIC, and our refined CSKB, Ascent++, have fundamental differences. First, regarding focused domains and sources of knowledge, AutoTOMIC relies on implicit knowledge from GPT-3 and focuses on event-centric knowledge. In contrast, Ascent++ extracts concept-centric knowledge explicitly mentioned in a large web corpus. Second, AutoTOMIC represents knowledge using the traditional SPO triple format, whereas Ascent++ uses a refined commonsense knowledge model to capture more expressive assertions. Third, as GPT-3 does not provide open access APIs, prompting it to build a large-scale CSKB incurs significant monetary costs [17] compared to extracting knowledge from freely available web crawls.

3 Knowledge Model

Existing CSKBs typically follow a triple-based data model, linking subjects via predicate phrases to object words or phrases. Typical examples from ConceptNet are (bus, Used-For, travel) and (bus, UsedFor, not taking the subway).

Few projects [6], [7] have attempted to sharpen such assertions by word sense disambiguation (WSD) [63], distinguishing, for example, buses on the road from computer buses. Likewise, only a few projects [8], [48], [50], [64] have tried to identify salient assertions against correct ones that are unspecific, atypical, or even misleading (e.g., buses used for avoiding the subway or used for enjoying the scenery). We extend this prevalent paradigm in two significant ways.

Expressive Subjects. CSK acquisition starts by collecting assertions for target subjects, usually single nouns. This has two handicaps: 1) it conflates different meanings for the same word, and 2) it misses out on refinements and variants of word senses. While some projects [6], [7] tried to use word sense disambiguation (WSD) to overcome the first issue, they were inherently limited by the underlying word-sense lexicons (WordNet or Wiktionary). Indeed, they mainly restrict themselves to single nouns. For example, phrases like “city bus”, “tourist bus”, “newborn elephant”, or “baby elephant” are not present.

Our approach to rectify this problem is twofold:

- First, when discovering source documents for the target subject, we compare the documents with their reference Wikipedia article and only retain documents with better similarity. This way, we can disentangle different senses, for example, of “bus” as in “public transport” and “network topology” themes.
- Second, when extracting candidates for assertions from the source documents, we also capture multi-word phrases as candidates for refined subjects. This way, we can acquire isa-like refinements to create subgroups of broader subjects (e.g., “school bus”, “city bus”, “tourist bus”, “circus elephant”, “elephant cow”, “domesticated elephant”), and also other kinds of aspects that are relevant to the general concept (e.g., “bus route”, “bus capacity”, “elephant tusks”, “elephant habitat” or “elephant keeper”).

Our notion of subgroups can be thought of as an inverse isa relation. It goes beyond traditional taxonomies by better coverage of multi-word composites (e.g., “circus elephant”, “school bus”).

This allows us to improve the representation of specialized assertions such as (circus elephants, catch, balls) and (school bus, transports, students).

Our notion of aspects includes part-whole relations (PartOf, MemberOf, SubstanceOf) [33], [42], [43], [65], but also further aspects that do not fall under the themes of hypernymy or meronymy (e.g., “bus accident”, “elephant habitat”). Note that, unlike single nouns, such compound phrases are rarely ambiguous, so we have crisp concepts without the need for explicit WSD.

Semantic Facets. For CSK, assertion validity often depends on specific temporal and spatial circumstances, e.g., elephants scare away lions only in Africa or bathe in rivers only during the daytime. Furthermore, assertions often become crisper by contextualization in terms of causes/effects and instruments (e.g., children ride the bus ...to go to school, circus elephants catch balls ...with their trunks).

To incorporate such information into an expressive model, we contextualize subject-predicate-object triples with semantic facets. To this end, we build on ideas from research on semantic role labeling (SRL) [51], [52], [53]. This line of research has been initially devised to fill hand-crafted frames (e.g., purchase) with values for frame-specific roles (e.g., buyer, goods, price, etc.). We start with 35 labels proposed in [54], a combination of those in the Illinois Curator SRL [52], and 22 hand-crafted ones derived from an analysis of semantic roles of prepositions in Wiktionary. As many of these are very special, we condense them into eight widely useful roles that are of relevance for CSK: four that qualify the validity of assertions (degree, location, temporal, other-quality), and four that capture other dimensions of context (cause, manner, purpose, transitive objects).

Quantitative Scoring. Previous works typically quantify the quality of assertions by a single numeric score. ConceptNet, for instance, scores its assertions essentially by the number of annotators that stated them (in most cases, 1). TupleKB employs a supervised model that predicts a [0-1] score capturing statement plausibility. With Ascent++, we want to empower downstream users to remain flexible in how to rank and use the data. We thus propose a scoring along two dimensions:

- Salience: This score captures how spontaneous an assertion comes to the human mind. For example, elephants being used for tourist rides is quite salient, while elephants sleeping at night is less so. Salience is important to understand which statements matter to humans (e.g., in conversational agents).
- Typicality: This dimension captures the degree to which an assertion applies to individual instances of
a concept, on a per-subject basis. For example, most elephants sleep in most nights, whereas only few elephants give tourists a ride. Typicality is important to understand which utterances make sense (e.g., in question answering).

Typicality and saliency are thus orthogonal dimensions, allowing to capture finer properties of commonsense assertions than just frequency or plausibility. Further dimensions could be considered [50], though we found plausibility not to be a dimension of high discriminative utility (implausible statements should rather not even enter CSKBs).

These design considerations lead us to the following knowledge model.

Let $C_0$ be a set of primary concepts of interest, which could be manually defined or taken from a dictionary. Subjects for assertions include all $s_0 \in C_0$ and judiciously selected multi-word phrases containing some $s_0$.

Subjects are interrelated by subgroup and aspect relations: each $s_0$ can be refined by a set of subgroup subjects denoted $sg(s_0)$ and by a set of aspect subjects denoted $asp(s_0)$. The overall set of subjects is $C := C_0 \cup sgC_0 \cup aspC_0$.

Definition [Commonsense Assertion]:
A commonsense assertion for $s \in C$ is a sextuple $(s, p, o, F, \pi, \theta)$ with single-noun or noun-phrase subject $s$, short phrases for predicate $p$ and object $o$, and a set $F$ of semantic facets. Each facet $(k, v) \in F$ is a key-value pair with one of eight possible keys $k$ and a short phrase as $v$.

Note that a single assertion can have multiple key-value pairs with the same key (e.g., different spatial phrases).

Furthermore, each assertion is accompanied by two $[0,1]$ scores: $\pi$ for saliency and $\theta$ for typicality.

4 Methodology

4.1 Architecture Overview

Design Considerations. CSK collection has three major design points: (i) the choice of sources, (ii) the choice of the extraction techniques, and (iii) the choice of cleaning or consolidating the extracting candidate assertions.

As sources, most prior works carefully selected high-quality input sources, including book n-grams [6], concept definitions in encyclopedic sources, and school text corpora about science [66]. These are often limiting factors in the KB coverage. Moreover, even seemingly clean texts like book n-grams come with surprisingly high noise and bias (cf. [67]). Focused queries for retrieving relevant web pages were used by [7], but the query formulations required non-negligible effort. Query auto-completion and question-answering forums were tapped by Quasimodo [8], [68]. While this gave access to highly salient assertions, it was, at the same time, adversely affected by heavily biased and sensational contents (e.g., search-engine auto-completion for “elephants eat” suggesting “...plastic” and “...poop”).

Our earlier version, ASCENT [19], also used search engines combined with refined search queries that use hypernyms for disambiguation. Then, each retrieved web page was compared with a reference document of its corresponding subject to make sure that the web page was not off-topic. This helped ASCENT collect very high-quality documents for various subjects such as animals (e.g., there are many web pages exclusively talking about animal facts). However, the method did not work well on subjects for which named instances are generally more prominent than the general concepts, such as “university” or “car”. For example, while for “elephant”, the method extracted 508 assertions with a frequency greater than one, for “university”, it found a mere 12. ASCENT++ rectifies this by directly using a huge corpus as an extraction source. This approach was also taken by ASER [26] and its derivative, TransOMCS [16], yet these approaches are recall-oriented and lack appropriate consolidation.

For the extraction techniques, choices range from co-occurrence- and pattern-based methods (e.g., [69]) and open information extraction (e.g., [7], [8]) to supervised learning for classification and sequence tagging. Co-occurrence works well for a few pre-specified, clearly distinguished predicates using distant seeds. Supervised extractors require training data for each predicate and thus have the same limitation. Recent approaches, therefore, prefer OpenIE techniques, and our extractors follow this trend, too.

For knowledge consolidation, early approaches kept all assertions from the ingest process (e.g., crowdsourcing [5]), whereas recent projects employed supervised classifiers or rankers for cleaning [7], [8], [48], [50], and also limited forms of clustering [7], [8] for canonicalization (taming semantic redundancy).

In ASCENT, we used contextual language models to cluster assertions of the same meanings and reinforce the frequency signals of those assertions. In the present ASCENT++, we apply the same technique with a state-of-the-art model for sentence embeddings, the SentenceTransformers model [20]. Furthermore, after clustering, we do a mapping from our open-schema CSKB to the well-established ConceptNet schema as it is favorable to many researchers (e.g., [11], [25], [70]). Finally, a heuristic-based cleaning approach is applied to eliminate other remaining noise in the resulting KB.

Approach. The new ASCENT++ methodology operates in two phases (Fig. 1):

1. Corpus processing
   1a. NLP pipeline: Running NLP pipeline for the input corpus to get NLP features of all sentences, particularly part-of-speech tags and dependency trees.
   1b. Faceted OpenIE: Running the ASCENT OpenIE system to get faceted assertions from the processed sentences.

2. Aggregation and consolidation: Each subject is processed separately in this phase.
   2a. Filtering: Performing a series of document and assertion filtering to get relevant and high-quality assertions for a given subject.
   2b. Clustering of retained assertions based on sentence embeddings.
   2c. Mapping from open assertions into ConceptNet schema.
   2d. Cleaning based on heuristics and a dictionary of unwanted assertions.
   2e. Ranking of assertions: Annotating assertions with complementary scores for typicality and saliency.

Both phases treat each document or subject independently and thus can be highly parallelized (see Section 5.2).
Since commonsense knowledge evolves rather slowly (e.g., smartphones emerging over the years), compared to encyclopedic knowledge where new entities and relations emerge on a daily basis, computing a large CSKB is a one-time endeavor with long-term value. Nonetheless, whenever new or updated inputs need to be processed, Phase 1 can be run incrementally on the new inputs only. Only steps 2b and 2e require re-loading previous statements (on a per-subject basis). Re-running these steps takes about half a day for a corpus like the C4 crawl [18]. Table 2 gives detailed run-times (see Section 5.2).

In the following, we discuss Phase 1 and steps 2a-2e of Phase 2 in separate subsections.

### 4.2 Corpus Processing (Phase 1)

Our previous version, Ascent, retrieves 500 web pages returned by the Bing Search APIs using hand-crafted search queries for each subject. Usually, not all of those pages would be available for the extraction steps due to crawling errors and another document filtering step in the Ascent pipeline. Thus, although this method gathered very high-quality sources, we had to sacrifice recall for several subjects.

We drop it to address these limitations of the web-search component in Ascent. Instead, Ascent++ can use any English corpus as its knowledge source. This enables the system to tap texts several orders of magnitude larger, making it easy to switch to any particular domain of interest.

**Phase 1a: NLP pipeline.** As we reuse the Ascent OpenIE approach [19], the NLP pipeline must consist of fundamental operations, including sentence splitting, tokenization, lemmatization, part-of-speech tagging, dependency parsing, and named entity recognition. The extractors will use all of these basic NLP features to output *faceted* OpenIE tuples.

**Phase 1b: Faceted OpenIE.** The Ascent OpenIE approach was built upon StuffIE [54], a rule-based system capable of extracting triples and semantic facets from English sentences. The core ideas of the approach are to consider each verb as a candidate predicate of an assertion and then identify subjects, objects, and facets via grammatical relations, so-called dependency paths. Subjects must be connected to the candidate predicate through subject-related dependency edges (nsubj, sbj, and csubj) or the adjectival clause edge (acl). Meanwhile, for objects, the respective edges include direct object (dobj), indirect object (iobj), and nominal modifier (nmod). Next, the semantic facets are identified through the following complements to the selected verb: adverb modifiers, prepositional and clausal complements. We also extended the original set of rules in StuffIE to better deal with conjuncts and adverb facets which helps to identify significantly more assertions and facets and improves the conciseness of the output tuples. For instance, given the sentence “elephants use their trunks to pick up objects and drink water”, our system can extract two assertions: (elephants, use, their trunks, purpose: pick up objects) and (elephants, use, their trunks, purpose: drink water).

Details on the dependency-pattern-based rules were previously described in Section 4.2 in [19].

Besides faceted OpenIE tuples, in Ascent, we also proposed heuristics to extract fine-grained subjects, i.e., subgroups and aspects, of a given primary concept.

In terms of *subgroups*, they could be sub-species in the case of animals, different types of the primary subject, or refer to the primary subject in different states, e.g., “African elephant”, “newborn elephant”, “male elephant”, “working elephant”.

For a primary subject $s_0$, we collect all noun chunks ending with $s_0$ or any of its WordNet lemmas as potential candidates. Then, we cluster these terms based on word2vec [71] embeddings. In addition, we leverage WordNet to distinguish antonyms with which the vector space embeddings typically struggle.

For *aspects*, given a primary subject $s_0$, its aspects are extracted from noun chunks collected from two sources: (i) possessive noun chunks, for example, “elephant’s diet” and “their diet” (with a resolution to “elephant’’); (ii) $s, p, noun chunk$ triples where $s$ is equal to $s_0$ or any of its WordNet lemmas, and $p$ is one of the following verb phrases: “have”, “contain”, “be assembled of” or “be composed of”.

The extracted subgroups and aspects are also normalized and cleaned in order to avoid spurious extractions or overly specific terms (see Section 4.2 in [19]).

### 4.3 Filtering (Phase 2a)

While the earlier search-engine-based document retrieval in Ascent helped for topical relevance (a core focus of search engines), using a general web corpus as an extraction source implies the presence of substantially more irrelevant content.

Given a subject $s$, one might first collect all OpenIE assertions whose subject is equal $s$ as candidate assertions and
then apply ranking or filtering techniques afterward. Similar approaches have been used in Quasimodo [8] and TupleKB [7]. Nevertheless, such post-hoc filtering misses out on broader context from the original documents. In Ascent++, we thus employ some filters first, at the document level, to decide which documents to use for candidate extraction at all. We only extract statements for a primary subject $s_0$ and its subgroups and aspects from a document $d$ if it passes the following filters:

1) Document $d$ is only used if it contains between 3 and 40 assertions for $s_0$. The rationale for filters in either direction is that if $s_0$ occurs too rarely, $d$ is more likely off-topic. On the other hand, if $s_0$ occurs too often, then $d$ may be a noisy document such as a machine-created shopping catalog or simply a crawling error.

2) Then, we compute the cosine similarity between the embeddings of $d$ and the Wikipedia article of the subject $s_0$. Document $d$ will be retained only if the similarity is higher than 0.6 (chosen based on tuning on withheld data). This way, we can deal with ambiguous subject terms like “bus”, which can be either a vehicle or a network topology.

After the previous filters, we come up with a list of documents for the subject $s_0$. Then, we collect all OpenIE assertions for $s_0$ and its subgroups and aspects from those documents. After aggregating the extracted assertions, we only retain those with a frequency of at least 3. That helps to remove noise and redundancy from the results.

### 4.4 Clustering (Phase 2b)

Natural language is rich in paraphrases. For example, “elephant eats grass” can also be written as “elephant feeds on grass” or “elephant consumes grass”.

Therefore, identifying and clustering such assertions is necessary to avoid redundancies and get better frequency signals for individual assertions.

For triple clustering, we use the hierarchical agglomerative clustering (HAC) algorithm along with LM-based embeddings to group semantically similar triples. Specifically, we use SentenceTransformers [20], a state-of-the-art architecture for sentence embeddings, to compute embeddings of CSK triples. First, given a triple, we concatenate its subject, predicate, and object. Next, we feed the whole string to SentenceTransformers to get its contextualized embeddings. Then, for each pair of triples, we compute the Euclidean distance between their normalized embeddings. These distances will be used as input for the HAC algorithm.

For facet clustering, we use average word2vec embeddings and the HAC algorithm, the same approach previously used in Ascent. Although more advanced language models such as PhraseBERT [72] could be used instead of word2vec, we found that, for such a limited number of candidate facets (usually less than 10 facets per assertion), word2vec embeddings already provide good performance.

The set of chosen hyper-parameters for the clustering algorithms will be presented in Section 5.3.

### 4.5 ConceptNet Mapping (Phase 2c)

**Motivation.** There are two main schools for knowledge representation in CSKBs: those relying on open predicates and those using a fixed set. Each has its strengths and challenges regarding expressiveness, redundancy, and usability. To bridge the two, we provide our Ascent++ KB in two variants: with open assertions and with canonicalized predicates. Open information extraction supplies the former; the module presented in this subsection normalizes open assertions into fixed relations.

Our fixed schema of choice is the established ConceptNet schema [5], from which we use the following 19 relations: AtLocation, CapableOf, Causes, CreatedBy, DefinedAs, Desires, HasA, HasPrerequisite, HasProperty, HasSubevent, IsA, MadeOf, MotivatedByGoal, PartOf, ReceivesAction, RelatedTo, SimilarTo, SymbolOf and UsedFor.

Mapping open triples to a fixed schema raises several challenges. In the most straightforward case, the subject and object from the open assertion can remain unchanged, and we only need to pick one of the fixed relations. For example, (elephant, lives in, the wild) can be mapped to (elephant, AtLocation, the wild).

In some cases, part of the relation and object can be moved, e.g., (elephant, is, a part of a herd) can be mapped to (elephant, PartOf, herd).

In other cases, part or all of the predicate is in the object, like in (circus elephant, catches, balls) that can be mapped to (circus elephant, CapableOf, catch balls).

Our normalization method consists of two steps: (i) first, we use a multi-class classifier to predict a fixed predicate for each open triple; (ii) second, we use a list of crafted rules to modify the object so that the new triple preserves the meaning of the original triple.

**Supervised Classifier and Rule-Based Disambiguation.** For the classification model, we fine-tune a RoBERTa model to predict one of the ConceptNet predicates given an input in the following format: [CLS] $S$ [SEP] $P$ $O$, whereas $S$, $P$, and $O$ are the subject, predicate, and object of the open triple. The contextualized embeddings of the [CLS] token will be fed to a fully connected layer to get the prediction. The training data for this model is constructed from ConceptNet triples. Specifically, for each ConceptNet predicate, we manually compiled a list of 1-6 open predicates with similar meanings. For instance, UsedFor is aligned with “be used for”, meanwhile CapableOf is aligned with “be capable of”, “be able to”, “can”, “could”, and an empty string. This way, we can automatically generate approximately 1.2M training examples.

Due to the nature of ConceptNet, the generated data is highly biased towards a few top predicates. The three most popular predicates are IsA (27.92%), AtLocation (20.24%) and CapableOf (13.76%). Meanwhile, the three least popular ones are MadeOf (0.20%), CreatedBy (0.06%) and SymbolOf (12 samples, less than 0.001%). This imbalance affects the predictions. The most important difficult case is distinguishing the three predicates IsA, HasProperty, and ReceivesAction, which can all be expressed with the open predicate word “be”. In the ConceptNet-based training data, the IsA relation (27.92% of the data) is dominant over the other two, HasProperty (3.07%) and ReceivesAction (2.20%). Therefore, the LM
occasionally classifies triples whose predicate is “be” incorrectly as IsA relations.

For this particular case, we have a post-processing step to adjust the predicted predicate: we only assign HasProperty to objects which are adjective phrases, ReceivesAction to objects which are verb phrases in passive form, and the rest are assigned to IsA. The IsA relation still contains considerable noise. Hence, later in the cleaning phase (see Section 4.6), we introduce heuristics to get high-quality assertions of this type.

An alternative could be to re-balance the training data by undersampling the frequent classes or adopting a loss function that gives different weights to different classes. While such generic techniques could be considered, our experience is mirrored in related projects on creating and curating high-quality KBs, where injecting a modest amount of expert knowledge is often the most effective solution [29], [73]. More advanced methods for relation alignment also exist [74], [75], [76], [77]. However, given that ConceptNet itself is imbalanced and has its peculiarities, our customized mapping is far superior to generic alignment methods.

Processing Objects. Once we get a predicted fixed-schema predicate, we have to produce an object for the normalized triple. The most common case when the object needs modifications is when the predicted predicate is CapableOf. In this case, if the open predicate is neither “be capable of”, “be able to”, “can” nor “could”, the new object will be the concatenation of the original predicate and object. In some cases, a part of the original object must be cut out as it overlaps with the ConceptNet predicate. Those predicates include PartOf and SymbolOf. Open triples corresponding to those predicates usually look like (elephant, is, part of a herd) or (elephant, is, symbol of strength) which should be canonicalized into (elephant, PartOf, herd) and (elephant, SymbolOf, strength), respectively.

Other rules deal with other predicates such as Desires, HasProperty, IsA, ReceivesAction, and UsedFor. In total, we have compiled 7 rules for object modification. We preserve the original object if a triple does not fall into one of these rules.

4.6 Cleaning (Phase 2D)

Noise can come from various sources, including OpenIE errors, too specific or general statements, nonsensical assertions, schema normalization errors, etc. Unlike supervised classifiers employed in other projects, our cleaning module is rule-based and thus highly scrutable. Ascent++.'s cleaning module consists of the following heuristics:

1) First, we compute the perplexity of all triples. To do so, we first turn the triples into sentences and then use an autoregressive model (GPT-2) to compute the perplexity of the sentences. Only triples with medium-to-low perplexity will be retained (in our experiments, we retained triples with perplexity less than 500).

2) The IsA triples produced by OpenIE are rich but rather noisy. Extracting IsA relations is a well-established research theme in entity typing and taxonomy construction and has already reached high-quality results. Therefore, we use the following cleaning algorithm. For each subject, we take the list of its IsA relations in ConceptNet and extract all head nouns of the objects. For example, from the ConceptNet triple (elephant, IsA, placental mammal), we extract the head noun “mammal”. Then, in Ascent++, we only retain the IsA assertions in which objects contain one of the trustworthy head nouns extracted earlier. This way, we not only get high-precision assertions but also better recall than ConceptNet. If a subject does not occur in ConceptNet, we remove all of its extracted IsA assertions, as in our observation, there are usually more noisy IsA assertions than valuable ones in the original extraction set. Note that, while ConceptNet is our source of choice here, any other existing resources that provide high-quality IsA relations can be used, for example, WordNet [39] or BabelNet [78].

3) Then, we manually constructed a dictionary of unwanted objects from our observations. For example, we removed URLs, pronouns, numbers, only stopwords, too general/specific phrases such as “make sure” or “this case”, and vague predicate-object pairs (e.g., MadeOf—“part”, HasProperty—“available”). We also eliminate ethnicity- and religion-related assertions to avoid potentially critical biases [79].

4) Finally, we only keep the 1,000 most frequent assertions per subject. For each assertion, we only keep its three most frequent facets. Cutting the tail this way improves the precision significantly but only minorly affects recall (see Section 6.1.3).

These filters are a pragmatic technique, and alternatives are conceivable. Some parts require manual work by a knowledge engineer. This holds particularly for the domain-specific dictionary filter. However, the dictionary is small, and a knowledge engineer can easily construct it within a day (at much lower energy consumption and carbon footprint than trying to automate everything computationally).

4.7 Ranking (Phase 2e)

Existing resources mainly provide unidimensional rankings of their assertions by either frequency (e.g., ConceptNet, Ascent) or supervised models trained to predict plausibility/typicality (TupleKB, Quasimodo, TransOMCS). However, these two dimensions are quite different, and we consider it important to differentiate their semantics.

Typicality states that an assertion holds for most instances of a concept. For example, elephants using their trunk is typical, whereas elephants drinking milk holds only for baby elephants. Saliency refers to the human perspective of whether an assertion is associated with a concept by most humans more or less on first thought. For example, “elephants have trunks” is salient, whereas “elephants pass by zebras” is not. In Ascent++, we make both dimensions first-class citizens of the CSKB and annotate each assertion with scores for both.

Saliency Ranking. For saliency, we rely on the reporting frequency of the assertions, which approximates very well how prominent an assertion is. We transform raw
### 5.1 Input Text Corpus and Subjects

Choosing the input text corpus is essential because large text corpora, especially those scraped from the web, often come with a large portion of noise and irrelevant contents. Our input corpus of choice is the Colossal Clean Crawled Corpus (i.e., the C4 corpus), created to train the T5 text-to-text Transformers model [18]. C4 was created by intensively cleaning the Common Crawl’s web crawl corpus. The filtering process includes deduplication, English-language text detection, removing pages containing source code, offensive language, or too few contents and lines, removing lines that did not end with a terminal punctuation mark, etc. That resulted in a large corpus of 750GB of text comprising reasonably clean and natural English text. The version we use, C4.EN, consists of 365M English articles. Each comes with its text, URL, and crawling timestamp. This amount of text enabled the pipeline to cover significantly more CSK assertions than the manually-built ConceptNet and any other automated CSK resource when limiting to top-100 assertions per subject (see Section 6.1.2). Our judicious rule-based approaches for extraction, filtering, and cleaning, as well as our unsupervised ranking, helped produce CSK assertions of higher saliency than any other automated CSK resource while maintaining a very high precision (see Section 6.1.1).

We executed the new Ascent++ pipeline for all primary subjects in Ascent (i.e., 10K popular concept-centric subjects taken from ConceptNet). For each primary subject, we took its top 10 most frequent aspects and top 10 most frequent subgroups previously extracted by Ascent as high-quality fine-grained subjects.

### 5.2 Processing Time and Output Size

In Table 2, we provide details on each step’s processing time and outputs in the Ascent++ pipeline.

The corpus processing phase (Phase 1) is executed once, independent of any choice of subjects. We use SpaCy, a popular Python-based NLP library, as our NLP pipeline. First, we ran SpaCy on the 365M documents from C4 in parallel on a cluster of 6,400 CPU cores (AMD EPYC 7702 64-core processors), which took 1.5 days to complete. Then, it took 20h for the OpenIE system to digest all 356M processed documents on the same CPU cluster. The result of this step is a set of 8B OpenIE assertions.

Next, the filtering process took around 2 hours and resulted in 165M assertions for the selected subjects, which account for 15M unique triples.

The clustering process, including precomputing embeddings and running the HAC algorithm, took around 10 hours. Embeddings were computed on a cluster of up to 150 GPUs (NVIDIA Quadro RTX 8000 GPUs) which took less than 2 hours. The HAC algorithm was run on the same CPU cluster, giving 7.5M clustered assertions. On average, two assertions form one cluster.

Next, the ConceptNet mapping module took around 30 minutes on the same GPU cluster. The rule-based cleaning step is the lightest component, taking less than 10 minutes on a personal laptop. It resulted in 2M assertions in the final Ascent++ CSKB, whose size is reported in Table 3.

---

1. https://huggingface.co/cardiffnlp/twitter-roberta-base-sentiment

2. https://spacy.io/
TABLE 2
Processing Time and Output Size of Each Step in the Pipeline

| #  | Phase                           | Processing time | Output                |
|----|---------------------------------|-----------------|-----------------------|
| 1a | NLP pipeline                    | 1.5 days        | 365M processed documents |
| 1b | Faceted OpenIE                  | 20 hours        | 8M OpenIE assertions  |
| 2a | Filtering                       | 2 hours         | 165M assertions → 15M unique triples |
| 2b | Clustering                      | 10 hours        | 7.5M assertion clusters |
| 2c | ConceptNet mapping              | 30 minutes      | 7.5M canonicalized assertions |
| 2d | Cleaning                        | 10 minutes      | 2M assertions         |
| 2e | Ranking                         | 2 hours         | 2M ranked assertions  |
| -  | Total                           | ~ 3 days        |                       |

6.1 Intrinsic Evaluation

To investigate RQ1, we instantiate quality with the standard notions of precision and recall, splitting precision further up into the dimensions of typicality and saliency, measuring this way the degree of truth and the degree of relevance of assertions (see Section 3 and also [8]).

Evaluation metrics. We employ three evaluation metrics:

1) typicality,
2) saliency,
3) relative recall.

Knowledge base construction is typically evaluated by precision and recall. Following earlier work [8], we split precision into two dimensions: typicality (the degree of truth) and saliency (how readily a statement is available to a human). These two dimensions are generally independent, as salient statements need not be typical, and vice versa. Furthermore, as the absolute recall is difficult to establish (e.g., there is no way to obtain a complete set of all assertions for what elephants are capable of doing), we use a relative recall metric, measuring the fraction of statements from a human-built resource, that are captured in the respective CSKB.

Compared Resources. We compare Ascent++ with its predecessor Ascent [19], as well as five other prominent resources:

1) ConceptNet [5],
2) TransOMCS [16],
3) TupleKB [7],
4) Quasimodo [8],
5) COMET-ATOMIC\textsuperscript{20} [25] (with a caveat, see below).

The ATOMIC\textsuperscript{20} [15], AutoTOMIC [17], and ASER [26] projects do not qualify for comparison as they do not contain concept-centric assertions. ATOMIC\textsuperscript{20} [25] has a portion for physical commonsense, but most of those assertions come directly from ConceptNet (except for the ObjectUse relation for which more human-annotated data was collected). Therefore, we do not include ATOMIC\textsuperscript{20} directly in this evaluation. Instead, we compare other CSKBs with a generative model trained on that resource, the COMET-ATOMIC\textsuperscript{20} model [25].

6.1.1 Precision: Typicality and Saliency

Evaluation Scheme. Unlike the precision of encyclopedic knowledge ("The Lion King" was either produced by Disney or not), the precision of CSK is generally not a binary concept, calling for more refined evaluation metrics.

Following the Quasimodo project [8], we assessed the typicality and saliency of triples. Given a CSK triple, we...
We performed two model \cite{25}. COMET is an autoregressive language model fine-tuned on existing CSK resources and can be used to predict possible objects of given subject-predicate pairs. Since COMET does not come with a materialized resource, we had to generate assertions ourselves. As there is no obvious stop criterion, we only evaluated the precision of top assertions but could not evaluate COMET's recall. We used the provided BART model\textsuperscript{5}, which was trained on ATOMIC\textsuperscript{20} (which also includes ConceptNet assertions). For each pair of subject and predicate, we asked COMET to predict top-5 objects. We used the same sampling processes and MTurk templates described above for typicality and saliency evaluation. The evaluation results of COMET are also included in Table \ref{tab:accuracy}. \textsc{Ascent++} clearly performs better than the generative model, even though both have seen comparable amounts of texts. Some examples of the top assertions in \textsc{Ascent++} and the ones generated by COMET-ATOMIC\textsuperscript{20} are shown in Table \ref{tab:atOMIC}. Although COMET has the flexibility of generating objects to any given subject-predicate pair, it still makes many incorrect predictions and produces notable redundancies. On the other hand, \textsc{Ascent++}, which collected OpenIE assertions from several sources and aggregated them through various steps such as filtering, clustering, and cleaning, produces more correct and complementary assertions.

### 6.1.2 Relative Recall

**Ground Truth.** Evaluating recall requires a notion of ground truth. We use a relative recall notion w.r.t. the statements contained in the CSLB property norm dataset \cite{82}, which consists of short human-written sentences about salient properties of general concepts. There are 22.6K sentences expressing properties of 638 concepts in the dataset. The CSLB dataset could also be considered a CSK resource. However, due to its limited size, we did not include it in the comparisons with other CSKBs. Instead, we used it as ground truth for evaluating relative recall and the mask prediction task (see Section 6.2).

**Assertion Matching Algorithm.** Since there are always different expressions of a CSK assertion in natural language, we allow soft matching between assertions in CSK resources and sentences in the ground-truth CSLB dataset. Specifically, for each CSLB sentence, we find the most similar assertion in the target CSK resource based on the cosine similarity between their embeddings (computed by SentenceTransformers \cite{20}). That assertion will be considered a true positive w.r.t the given CSLB fact only if their cosine similarity is greater than or equal to a predefined threshold $\tau$. For example, if $\tau = 1.0$, only exact-match assertions are considered true positives. When lowering $\tau$, we can match, e.g., “elephant eats grass” and “elephant feeds on grasses”.

**Results.** The relative recall evaluation results are shown in the right part of Table \ref{tab:accuracy}, where we once show relative recall using the top-100 assertions per subject and once all. We report results at three similarity thresholds: $\tau = 0.96$, $\tau = 0.98$ and $\tau = 1.0$.

We find that \textsc{Ascent++} yields the highest relative recall among all automated resources of the same size (“top-100” columns in Table \ref{tab:accuracy}), outperforming the next best KB, Quasimodo, by 14-15% in relative recall. The gap to the manually built ConceptNet is even larger, where \textsc{Ascent++} achieves 2 to 5 times higher relative recall, depending on the similarity threshold.

Considering all statements from each resource, Quasimodo and TransOMCS appear to have a slight edge; yet

---

\textsuperscript{5} https://github.com/allenai/comet-atomic-2020/tree/master/models/comet_atomic2020_bart
this is only due to the precision-oriented thresholding of Ascent++ (2M assertions versus 18.5M for TransOMCS and 6.3M for Quasimodo). Without the cleaning phase (see Section 4.6), the unfiltered Ascent++ variant would be the size-wise better point of comparison: with a size of 7.5M assertions, it achieves relative recall scores of 22.13%, 18.02%, and 9.37% for the three thresholds \( \tau \in \{0.96, 0.98, 1.0\} \), respectively. We also sampled 500 triples from each of the three KBs (unfiltered-Ascent++, Quasimodo and TransOMCS) for another typicality evaluation. The obtained typicality scores are: 82.2% for unfiltered-Ascent++, 56.4% for Quasimodo and 51.2% for TransOMCS. Hence, this variant of Ascent++ outperforms TransOMCS on both precision and recall, and it is better than Quasimodo in precision while having similar recall. Moreover, in Section 6.1.3, we will show that when increasing the size of Ascent++ to reach that of Ascent, we still reach competitive typicality scores. This gives us the flexibility to tune for either precision or recall, by adjusting the cleaning phase.

6.1.3 Ascent++ versus Ascent

To compare Ascent++ and its predecessor, Ascent, in addition to saliency@10 and typicality-all for the 200 common subjects, as well as the relative recall (see Table 5), we perform three more evaluations: typicality@10, typicality-random and saliency-random (see Table 7).

For typicality@10, we perform the same sampling process as for saliency@10; the only difference is that Ascent++ assertions are now sorted by typicality score instead of saliency score.

For typicality-random and saliency-random, a fair comparison must consider the different sizes of the two

| TABLE 4
| Crowdsourcing Question Templates for Typicality and Saliency Evaluation of CSK Assertions |
| --- |
| **Dimension** | **Question** | **Options** |
| Typicality | Is this a correct assertion about `<subject>`? | Always/Often - the knowledge assertion presented is always or often true |
| | | Sometimes/Likely - it is sometimes or likely true |
| | | Farfetched/Never - it is false or farfetched at best |
| | | Invalid - it is invalid or makes no sense |
| Saliency | Imagine you have 2 minutes time to explain a kid about `<subject>`, would you mention the following information? | Absolutely - the information is very interesting/important for that concept |
| | | Probably - it is quite good to know |
| | | Maybe not - it is not interesting or too obvious/uninteresting/boring |
| | | Definitely not - it is completely irrelevant/unimportant/wrong or makes no sense |

| TABLE 5
| Intrinsic Evaluation Results of Different CSK Resources |
| --- |
| **CSK resource** | **Precision (%)** | **Relative recall (%)** | **Size** |
| | Saliency@10 | Typicality (all) | Top-100 assertions/subject | All assertions | #spo |
| Crowdsourced | | | | | |
| ConceptNet [5] | 79.2 | 96.0 | 5.29 | 3.39 | 1.10 | 5.47 | 3.53 | 1.13 | 0.5M |
| Generative | | | | | |
| COMET-ATOMIC\(^{20}\) [25] | 55.2 | 78.9 | - | - | - | - | - | - |
| Extractive | | | | | |
| TransOMCS [16] | 40.4 | 51.4 | 4.04 | 3.24 | 1.65 | 19.70 | 16.47 | 9.06 | 18.5M |
| TupleKB [7] | 36.0 | 92.0 | 3.57 | 1.99 | 0.41 | 4.32 | 2.49 | 0.58 | 0.3M |
| Quasimodo [8] | 38.8 | 67.8 | 11.05 | 9.47 | 5.17 | 21.87 | 19.36 | 10.88 | 6.3M |
| Ascent [19] | 60.0 | 79.2 | 9.60 | 7.02 | 3.17 | 17.09 | 12.62 | 5.82 | 8.6M |
| Ascent++ | **68.8** | **88.4** | **12.70** | **10.70** | **5.90** | **17.54** | **14.64** | **7.95** | **2.0M** |

| TABLE 6
| Top-5 Assertions of Selected Subject-Predicate Pairs in Ascent++ and COMET-ATOMIC\(^{20}\) [25] |
| --- |
| **Subject-Predicate** | **Ascent++** | **COMET-ATOMIC\(^{20}\)** |
| elephant - CapableOf | - perform trick | - climb tree |
| | - eat grass | - walk on land |
| | - eat fruit | - climb tree trunk |
| | - become agitated | - walk on tree |
| | - give ride | - eat elephant |
| beer - Made(Up)Of | - hop | - beer |
| | - water | - alcohol |
| | - barley | - drunk |
| | - yeast | - drinking |
| | - grain | - drink |
| laptop - UsedFor/ ObjectUse | - work | - browse the internet |
| | - gaming | - use as a coaster |
| | - officew work | - play games on |
| | - email | - use as a weapon |
| | - social media | - browse the web |
Masked prediction is perhaps the best-researched problem, coming with the advantage of allowing automated evaluation, although automated evaluation may unfairly discount sensible alternative answers. Also, masked prediction is limited to single tokens. Autoregressive LM-based generations circumvent this restriction, although they necessitate human annotations and can be prone to evasive answers. They are thus well complemented by extractive answering schemes, limiting the language models’ abstraction abilities but providing the cleanest way to evaluate the context alone.

Models. Following standard usage, we use RoBERTa-large [80] for masked prediction, GPT-3 [21] for the generative setup, and ALBERT-xxlarge [85], fine-tuned on SQuAD 2.0, for span prediction.

Context Retrieval Method. Given a query, we use sentence embeddings to pull out relevant assertions from a CSKB. Using the SentenceTransformers model msmarco-distillbert-base-v3, we compute embeddings of the given query and all lexicalized triples in the CSKBs. Then we use cosine similarity to select the top 5 most similar triples to the query as context.

Task Construction. Previous work has generated masked sentences based on templates from ConceptNet triples [58]. However, the resulting sentences are often unnatural, following the idiosyncrasies of the ConceptNet data model.

Therefore, we created a new dataset of natural commonsense sentences for masked prediction that is based on an independent human-annotated dataset, the aforementioned CSLB dataset [82], and consists of 19.6K masked sentences [19].

For the generative and extractive settings, we used the Google Search auto-completion functionality to collect commonsense questions about common subjects by feeding the API with six prefixes: “what/when/where are/do <subject> ...”. That process returned 8,098 auto-completed queries.

Next, we drew samples from the query set, then manually removed jokes and other noise (e.g., “where do cows go for entertainment”), obtaining 100 questions for evaluation.

Evaluation Scheme. For commonsense topics, questions often have multiple valid answers. Additionally, given that answers in our generative and extractive QA settings are very open, creating an automated evaluation is difficult. We, therefore, use human judgments for evaluating all settings except masked prediction. Specifically, given a question and set of answers, we ask human annotators to assess each answer based on two dimensions, correctness, and informativeness, each on a 4-point Likert scale from 0 (lowest) to 3 (highest) (see Table 9 for the question templates given to the annotators). Correctness indicates whether an answer holds true. Informativeness reflects that the information conveyed by an answer is helpful.

An answer could be correct and uninformative at the same time. For example, given the question “What do elephants use their trunk for?”, the answers “For breathing,” and “To suck up water.” are both correct and informative. However, “To do things.” is a correct answer but not

---

### TABLE 7

| CSKB           | Typicality@10 Random | Typicality@10 Random | Saliency Random |
|----------------|----------------------|----------------------|-----------------|
| Ascent [19]    | 87.6                 | 65.6                 | 40.6            |
| Ascent++       | 93.6                 | 82.0                 | 44.6            |

---

6. https://www.sbert.net/docs/pretrained-models/msmarco-v3.html
informative at all. On the other hand, if an answer is incorrect, it should automatically be uninformative.

Three annotators evaluate each question in Amazon MTurk with the same qualification requirements as in Section 6.1. We use the mean precision at $k (P@k)$ metric for evaluating masked prediction, following [58].

Results. The evaluation results are shown in Table 10. For masked prediction, all CSKBs contribute useful contexts that substantially improve the quality of LM responses. A**SCENT++, along with the only manually-constructed KB, ConceptNet, statistically significantly outperforms all other KBs at every threshold $k$ (all p-values of paired t-test below 0.05).

For autoregressive LM-based QA, markedly, we find that GPT-3 performs on average better without any context. However, in combination with A**SCENT++, it still performs better than with any other CSKB. More research is needed to design methods to pull relevant context from CSKBs and decide when to use it in LM-based QA and when to rely on the LM’s knowledge alone.

For span prediction, where answers come directly from retrieved contexts, A**SCENT++ also outperforms all other competitors. A**SCENT++ obtained statistically significant gains over Quasimodo, TupleKB, and TransOMCS on both metrics, and over ConceptNet on correctness. This indicates that our A**SCENT++ assertions have high quality compared to others.

6.3 Evaluation of Facets
To answer RQ3, we evaluate facets both intrinsically and extrinsically.

Intrinsic Evaluation. As there are no existing CSKBs coming with facets, we provide comparisons with a strong LM baseline, GPT-2 [57]. First, we randomly drew 300 assertions along with their top-1 facets from our KB. Next, we translate each statement into a sentence prefix and ask GPT-2 to fill in the remaining words to complete the sentence. For example, given the quadruple $(elephant, use, their trunks, PURPOSE: to suck up water)$, the sentence prefix will be “Elephants use their trunk to” and for this, GPT-2’s continuation is “to move around.” Then, each sentence prefix along with two answers (from A**SCENT++ and GPT-2) were shown to a human annotator (without knowing the source of the answers) who annotated if each answer was correct/incorrect and informative/uninformative, following similar metrics used for the QA evaluation in Section 6.2.

| Context          | Sample output                  | Sample output was given by RoBERTa (for MP), GPT-3 (for AR) and ALBERT (for SP). |
|------------------|--------------------------------|--------------------------------------------------------------------------------|

| TABLE 9          | Crowdsourcing Question Templates for Evaluation of the AR and SP Settings for Commonsense QA |
|------------------|------------------------------------------------------------------------------------------------|
| Dimension        | Question Options                                                                         |
| Correctness      | 3 - Always/Often - the answer is always or often true                                    |
|                  | 2 - Sometimes/Likely - it is sometimes or likely true                                     |
|                  | 1 - Farfetched/Never - it is false or farfetched at best                                 |
|                  | 0 - Invalid - it is invalid or makes no sense                                            |
| Informativeness  | 3 - Highly - the answer provides very useful knowledge w.r.t the question                 |
|                  | 2 - Moderately - the answer is moderately useful                                         |
|                  | 1 - Slightly - the answer is slightly useful                                             |
|                  | 0 - Not at all - the answer is too general or makes no sense                              |

| TABLE 10         | QA Evaluation Results                                                                 |
|------------------|----------------------------------------------------------------------------------------|
| Context          | MP | AR | SP |
|------------------|----|----|----|
| No context       | 8.10 | 27.08 | 14.41 |
| ConceptNet       | 11.61 | 24.76 | 22.75 |
| TransOMCS        | 12.11 | 27.03 | 13.30 |
| TupleKB          | 12.11 | 27.03 | 13.30 |
| Quasimodo        | 11.61 | 24.76 | 22.75 |
| A**SCENT         | 11.95 | 24.70 | 29.70 |
| A**SCENT++       | 13.30 | 27.03 | 32.90 |
| Metrics: $P@k$ - mean precision at $k$ (%), $C$ - correctness ([0, 3]), $I$ - informativeness ([0, 3]). |
The results are reported in Table 11. Ascent++ achieves 70.10% correctness and 54.15% informativeness, both significantly better than the values for the GPT-2 model.

**Extrinsic Evaluation.** We reused the three question answering tasks from Section 6.2. The results are shown in Table 12. Expanding the Ascent++ triples with facets gives a consistent improvement in four of five evaluation metrics (precision at one in MP, informativeness in AR and SP, and correctness in SP), with the most prominent effect being observed for span prediction (8.6% and 9.3% relative improvements over the no-facet context in informativeness and correctness, respectively).

### 7 Conclusion

This paper presented Ascent++, a methodology to extract and semantically organize advanced commonsense knowledge from large-scale web contents. Our refined knowledge representation allowed us to identify considerably more informative assertions, overcoming the limitations of prior works. The techniques for filtering, aggregating, and consolidating extracted triples show that CSK extraction from broad web content is feasible at scale, with both high precision and high recall. Intrinsically and extrinsically evaluated triplets confirmed that the resulting CSKB is a significant advance over existing CSK collections and provides an edge over recent language-model-based approaches. Code, data, and a web interface are accessible at https://ascentpp.mpi-inf.mpg.de.

### References

[1] J. McCarthy, *Programs With Common Sense*. Cambridge, MA, USA: RLE and MIT Computation center, 1960.

[2] E. A. Feigenbaum, “Knowledge engineering: The applied side of artificial intelligence,” *Ann. New York Acad. Sci.*, vol. 426, no. 1, pp. 91–107, 1984.

[3] D. B. Lenat, “CYC: A large-scale investment in knowledge infrastructure,” *Commun. ACM*, vol. 38, pp. 33–38, 1995.

[4] “The OpenCyc Platform,” Accessed: May 16, 2022. [Online]. Available: https://web.archive.org/web/20160330064354/http://www.opencyc.org/

[5] R. Speer and C. Havasi, “ConceptNet 5: A large semantic network for relational knowledge,” *Theory and Applications of Natural Language Processing*, Berlin, Germany: Springer, 2012.

[6] N. Tandon, G. de Melo, F. M. Suchanek, and G. Weikum, “WebChild: Harvesting and organizing commonsense knowledge from the web,” in *Proc. Int. Conf. Web Search Data Mining*, 2014, pp. 523–532.

[7] B. D. Mishra, N. Tandon, and P. Clark, “Domain-targeted, high precision knowledge extraction and correction,” *Trans. Assoc. Comput. Linguistics*, vol. 5, pp. 233–246, 2017.

[8] J. Romero, S. Razniewski, K. Pal, J. Z. Pan, A. Sakhadeo, and G. Weikum, “Commonsense properties from query logs and question answering forums,” in *Proc. Conf. Inf. Knowl. Manage.*, 2019, pp. 1411–1420.

[9] K. Panton et al., “Common sense reasoning - from CYC to intelligent assistant,” in *Proc. Ambient Intelligence in Everyday*, Berlin, Germany: Springer, 2006, pp. 1–31.

[10] H. Lin, L. Sun, and X. Han, “Reasoning with heterogeneous knowledge for commonsense machine comprehension,” in *Proc. Conf. Empir. Methods Natural Lang. Process.*, 2017, pp. 2032–2043.

[11] B. Y. Lin, X. Chen, J. Chen, and X. Ren, “Kagset: Knowledge-aware graph networks for commonsense reasoning,” in *Proc. Conf. Empir. Methods Natural Lang. Process. Joint Conf. Natural Lang. Process.*, 2019, pp. 2829–2839.

[12] J. Xia, C. Wu, and M. Yan, “Incorporating relation knowledge into commonsense reading comprehension with multi-task learning,” in *Proc. Conf. Inf. Knowl. Manage.*, 2019, pp. 2393–2396.

[13] B. Y. Lin et al., “CommonGen: A constrained text generation challenge for generative commonsense reasoning,” in *Proc. Conf. Empir. Methods Natural Lang. Process.*, 2020, pp. 1823–1840.

[14] F. Ilievski, P. Szekely, and D. Schwabe, “Commonsense knowledge in Wikidata,” in *Proc. Wikidata Workshop*, 2020.

[15] M. Sap et al., “Atomic: An atlas of machine commonsense for if-then reasoning,” in *Proc. Conf. Assoc. Advance. Artif. Intell.*, 2018, pp. 3027–3035.

[16] H. Zhang, D. Khashabi, Y. Song, and D. Roth, “TransOMCS: From linguistic graphs to commonsense knowledge,” in *Proc. Int. Joint Conf. Artif. Intell.*, 2020, pp. 2284–2290.

[17] P. West et al., “Symbolic knowledge distillation: From general language models to commonsense models,” in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, 2022, pp. 4602–4625.

[18] C. Raiffel et al., “Exploring the limits of transfer learning with a unified text-to-text transformer,” *Jo. Mach. Learn. Res.*, vol. 21, pp. 1–67, 2020.

[19] T.-P. Nguyen, S. Razniewski, and G. Weikum, “Advanced semantics for commonsense knowledge extraction,” in *Proc. Int. World Wide Web Conf.*, 2021, pp. 2636–2647.

[20] N. Reimers and I. Gurevych, “Sentence-BERT: Sentence embeddings using siamese BERT-networks,” in *Proc. Conf. Empir. Methods Natural Lang. Process.*, 2019, pp. 3982–3992.

[21] T. B. Brown et al., “Language models are few-shot learners,” in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 1877–1901.

[22] P. Singh, T. Lin, E. Mueller, G. Lim, T. Perkins, and W. L. Zhu, “Open mind common sense: Knowledge acquisition from the general public,” in *Proc. OTM Confederated Int. Conf.*, 2002, pp. 1223–1237.

[23] H. Liu and P. Singh, “ConceptNet–A practical commonsense reasoning tool-kit,” *BT Technol. J.*, vol. 22, pp. 211–226, 2004.

[24] J. Gordon, B. V. Durme, and L. K. Schubert, “Learning from the web: Extracting general world knowledge from noisy text,” in *Proc. Conf. Assoc. Advance. Artif. Intell. Workshops*, 2010, pp. 10–15.

[25] J. D. Hwang et al., “(Comet)–Atomic 2020: On symbolic and neural commonsense knowledge graphs,” in *Proc. Conf. Assoc. Advance. Artif. Intell.*, 2021, pp. 6384–6392.

[26] H. Zhang, X. Liu, H. Pan, Y. Song, and C. W.-K. Leung, “ASER: A large-scale eventuality knowledge graph,” in *Proc. Int. World Wide Web Conf.*, 2020, pp. 201–211.

[27] H. Zhang et al., “ASER: Towards large-scale commonsense knowledge acquisition via higher-order selectional preference over eventualities,” *Artif. Intell.*, 2022, Art. no. 103740.

[28] F. Ilievski, P. Szekely, and B. Zhang, “Cskg: The commonsense knowledge graph,” in *Proc. Eur. Semantic Web Conf.*, 2021, pp. 680–696.

[29] S. Bhakthavatsalam, C. Anastasiades, and P. Clark, “Genericskb: A knowledge base of generic statements,” 2020, arXiv:2005.00660.

[30] T.-P. Nguyen, S. Razniewski, and G. Weikum, “Inside ASCENT: Exploring a deep commonsense knowledge base and its usage in question answering,” in *Proc. Assoc. Comput. Linguistics Syst. Demonstrations*, 2021, pp. 40–47.
Tuan-Phong Nguyen received the BSc degree in computer science from Vietnam National University Hanoi, in 2016 and the MSc degree in computer science from Saarland University, in 2020. He is currently working toward the PhD degree with the Department of Databases and Information Systems, the Max Planck Institute for Informatics and the International Max Planck Research School on Trustworthy Computing. His research interests span Commonsense Knowledge Base Construction, Information Extraction and Natural Language Processing.

Simon Razniewski received the Diploma (MSc) degree from TU Dresden, in 2010 and the PhD degree from the Free University of Bozen-Bolzano, in 2014. He is senior researcher with the Max Planck Institute for Informatics in Saarbrücken, Germany, where he heads the Knowledge Base Construction and Quality area. He was previously assistant professor with the Free University of Bozen Bolzano (2014-2017). He spent time as visitor with the Max-Planck Institute for Informatics (2016), the University of Queensland (2015), AT&T Labs-Research (2013), the University of California, San Diego (2012), and has previous industrial experience from Globalfoundries (2010) and Siemens IT (2009).

Julien Romero received an engineering diploma from Telecom Paris, the master of science degree from the ETH, Zürich, Switzerland, in 2020, and the PhD degree in computer science from Telecom Paris, France, in 2020-2021. He is currently an associate professor with Telecom SudParis, France, in the SAMOVAR lab. Before that, he made a postdoc with the Max Planck Institute of Informatics. His research interests cover knowledge bases, information extraction, natural language processing, and databases.

Gerhard Weikum is a scientific director with the Max Planck Institute for Informatics in Saarbrücken, Germany, and an adjunct professor with Saarland University. He coauthored a comprehensive textbook on transactional systems, received the VLDB Test of Time Award for his work on automatic database tuning, and is one of the creators of the YAGO knowledge base which received the WWW Test of Time Award. He is an ACM Fellow and received the ACM SIGMOD Edgar F. Codd Innovations Award.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/csdl.