Wikimarks: Harvesting Relevance Benchmarks from Wikipedia

Laura Dietz
University of New Hampshire, USA
dietz@cs.unh.edu

Shubham Chatterjee
University of New Hampshire, USA
Shubham.Chatterjee@unh.edu

Connor Lennox
University of New Hampshire, USA
Connor.Lennox@unh.edu

Sumanta Kashyapi
University of New Hampshire, USA
Sumanta.Kashyapi@unh.edu

Pooja Oza
University of New Hampshire, USA
PoojaHimanshu.Oza@unh.edu

Ben Gamari
Well-typed LLP, UK
ben@well-typed.com

ABSTRACT

We provide a resource for automatically harvesting relevance benchmarks from Wikipedia – which we refer to as "Wikimarks" to differentiate them from manually created benchmarks. Unlike simulated benchmarks, they are based on manual annotations of Wikipedia authors. Studies on the TREC Complex Answer Retrieval track demonstrated that leaderboards under Wikimarks and manually annotated benchmarks are very similar. Because of their availability, Wikimarks can fill an important need for Information Retrieval research.

We provide a meta-resource to harvest Wikimarks for several information retrieval tasks across different languages: paragraph retrieval, entity ranking, query-specific clustering, outline prediction, and relevant entity linking and many more. In addition, we provide example Wikimarks for English, Simple English, and Japanese derived from the 01/01/2022 Wikipedia dump.

Resource available: **https://trema-unh.github.io/wikimarks/**

CCS CONCEPTS

• **Information systems → Evaluation of retrieval results**.

KEYWORDS

test collections; relevant entity linking; query-specific clustering

ACM Reference Format:
Laura Dietz, Shubham Chatterjee, Connor Lennox, Sumanta Kashyapi, Pooja Oza, and Ben Gamari. 2022. Wikimarks: Harvesting Relevance Benchmarks from Wikipedia. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22), July 11–15, 2022, Madrid, Spain.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3477495.3531731

1 INTRODUCTION

Many information retrieval benchmarks have complicated licensing requirements, potentially impacting the dissemination and reproducibility of approaches. Wikipedias are freely available under a Creative Commons Share-Alike license, which explicitly allows the redistribution of derived data sets.

With this resource paper, we provide a conversion pipeline for deriving fully automated test collections from Wikipedia for several information retrieval tasks. Our approach is based on the assumption that Wikipedia pages were written to answer imagined information needs expressed in the page title (and/or headings). While many Wikipedia pages center on people and locations rather than general information needs, we focus on the remainder of pages which cover a wide range of topics of general interest, such as "Geomagnetic Reversal", "Wildlife Management", and "Superfood".

Each of these page topics are interpreted within an information retrieval scenario where a user with limited prior knowledge is seeking for a general overview on the topic. A system for such information needs might retrieve, cluster, and organize information into a single integrated response. Our idea is to provide benchmarks to study each of those tasks across a shared set of topics.

1.1 Query-specific Clustering and NLP

Traditionally, approaches for query-specific document rankings are studied independently of approaches for information extraction and data mining. However, in order to best support information seeking behaviors the retrieval stage needs to be working in concert with clustering, classification, and tagging stages. To remedy this, we suggest to study several well-known NLP and data mining tasks in the context of producing results that are relevant for the given query. We believe significant quality improvements for downstream applications are possible.

Our Wikimarks approach allows to study these tasks jointly by providing benchmarks for different tasks on the same set of topics.

Query-specific clustering. The canonical approach to search result clustering is to apply K-means or Agglomerative clustering the set of top k documents in a ranking. Note that the query only affects the resulting clustering via the search results, but the query is not directly included in the clustering phase. Instead, Kashyapi and Dietz [18] propose the following variant on clustering:

Task: Given a query q and a set of relevant candidate passages $\mathcal{P}$, the goal is to produce k *relevant* clusters $C_1\, C_2, \ldots C_k$ with each cluster $C_i \subseteq \mathcal{P}$ covering the passages $\mathcal{P}$ with non-overlapping sets. Here *relevant* means that the passages $p \in C_i$ from the same cluster share a subtopic that is relevant for the query q.

Kashyapi et al. demonstrate significant performance improvements on search-result clustering whenever the query is taken into consideration. The motivation for their work is that, depending on the user's information need, some clusterings are more relevant than others—regardless of changes to the result set. For example, if a user wants to know about the horseshoe crab's geographic distribution, then clustering results by country is appropriate, but when the user wants to know about the horseshoe crab's habitat, dividing results into different landforms such as shore, bay, and open sea is more useful. Following their ideas [18], we provides a benchmark for query-specific clustering derived from Wikipedia.

Relevant entity linking. Given a text passage, the task of entity linking is to detect mentions of knowledge base entities. Typically, the goal is to annotate *all* entity mentions. Entity linking is used mainly for two purposes: To present the results in a browsable interface to the user, or leverage the entity knowledge to improve the ranking performance. We argue that in neither case is it helpful to annotate *all* entity mentions — we rather would prefer to annotate only those entities that are central and relevant in the context of the query and the text passage. We propose the following variant on the entity linking task:

Task: Given a text passage p and query q, identify the set of relevant entities $\mathcal{E}$ mentioned in the passage. For every entity $e \in \mathcal{E}$ provide its knowledge base identifier and locate the offsets (b, e) of text spans where the entity is mentioned.

Previously, training data for the entity linking task was derived from Wikipedia pages [10, 21]. However, Wikipedia's editorial policy suggests to only annotate entities that are relevant for the topic of the article. Commonly this is noted as a weakness of Wikipedia-derived entity linking benchmarks, however we argue that it is a strength: it allows us to study the task of linking *only relevant* entities and to take the search query into consideration when making entity linking decisions.

Using the hyperlinks on Wikipedia articles, we can derive benchmarks for several entity-oriented tasks, such as relevant entity linking, entity retrieval, or query-specific entity clustering.

Multi-lingual and cross-language tasks. Wikipedias are available in many different languages. This allows us to study retrieval tasks in different languages with little extra effort.

While each Wikipedia might emphasize different topics based on different cultures, a significant number of topics[1] have corresponding pages across several Wikipedias, identified by a shared Wikidata QID. By providing cross-language benchmarks on different inter-related tasks, our resource provides avenues for novel cross-language approaches. For example, using our benchmark, one could learn how to retrieve and organize materials for queries in one low-resource language, based on the organization of similar articles in other languages.

1.2 Wikimarks

Our approach for harvesting relevance benchmarks is based on the assumption that for information needs derived from Wikipedia titles, the actual content of the Wikipedia article constitutes a relevant response. Based on this assumption, we can train and evaluate several information-centric tasks: passage retrieval as well as query-specific subtopic clustering and relevant entity linking.

[1]Especially pages denoted as "vital".

We refer to our automatically harvested relevance benchmarks as "*Wikimarks*" to differentiate them from manually created benchmarks. While our prior studies [13] found that evaluation results based on *Wikimarks* strongly correlate with results on manually created benchmarks, we advocate to use *Wikimarks* as a training criterion and early evaluation paradigm—to be complemented with manually created test collections.

Contributions. This resource provides the software and customizable toolchain for (1) converting any Wikipedia dump to an easily machine-accessible format and (2) harvesting a variety of *Wikimarks* from the dump.

The toolchain was recently extended to (3) provide multilingual support and (4) includes Wikidata[2] "Q" identifiers (QIDs) of pages, which are stable across time and language. As machine-readable interchange formats for the converted Wikipedia dump, the tool chain now supports (5) the JSON-lines[3] format (JSON-L) which allows for easy inspection of the data model in a text editor. This is in addition to the Concise Binary Object Representation[4] (CBOR) as defined in Internet Standard RFC 8949.

We provide (6) converted Wikipedia dumps from English, Simple English, and Japanese Wikipedia, dating to January 1st, 2022. As concrete examples, we (7) provide concrete *Wikimarks* for passage retrieval, entity retrieval, query-specific clustering, and relevant entity linking for each of these three dumps.

2 RELATED WORK

A wide range of manual test collections are widely used in the information retrieval community. However, the manual effort associated with their creation often prohibits the study of novel information seeking tasks, such as query-specific variants of clustering or entity linking, or outline generation for novel search interfaces.

Several approaches to semi-automatic support in test collection creation are discussed, including active learning for annotation and evaluation metrics that correct for resulting biases [9, 17, 28, 29, –*inter alia*]. While these significantly reduce the cost, they still rely on manual annotations.

In this work, we provide a pipeline, dump, and methods for fully automated test collection creation. These are not intended to replace manual test collections, but to complement them.

2.1 Fully Automatic Test Collections

Approaches for fully automatic benchmark creation have been discussed in the community. A popular approaches have been suggested by Azzopardi et al. [4], Berendsen et al. [7], where a artificial queries are simulated by selecting terms that maximize the probability of discriminating between the relevant and non-relevant document set. An open question is how to ensure that queries represent realistic information needs.

Alternatively, metadata of articles can be exploited, such as using anchor text [3], metadata of scientific articles about method, classification, and control [6], categories in the Open Directory Project [5], or glosses in Freebase [12]. The resource we provide in this paper is another example of this approach.

[2]https://www.wikidata.org
[3]https://jsonlines.org/
[4]https://www.rfc-editor.org/rfc/rfc8949.html

Within the TREC Complex Answer Retrieval track, both automatic and manual test collections were used. A study [13] compared the official leaderboard of submitted approaches using the manual ground truth with the leaderboard under the automatic test collection. Both leaderboards are very similar, suggesting that the automatic ground truth is useful for early method development. In this resource, we provide code for creating the automatic test collection and significantly extend this approach.

2.2 Related Resources in Related Fields

Several research communities are relying on Wikipedia to derive collections and benchmarks.

To study the fact extraction and verification task, the FEVER shared task[5] [1] uses a large collection of manually verified claims which are annotated with evidence in the form of sentences or table cells on Wikipedia pages.

The summarization community evaluate predicted summaries to whether they are similar to the lead text (i.e., text above the first section heading) given the remainder of a Wikipedia article. Ghalandari et al. [14] derive summarization dataset from the Wikipedia event portal. Perez-Beltrachini and Lapata [22] suggest similar benchmarks for cross-lingual summarization across different language versions of the same article. In open-domain aspect-based summarization, the task is to provide targeted summaries of a document from different perspectives. The WikiAsp dataset [15] derives these perspectives from section headings of Wikipedia articles.

In SECTOR [2], a benchmark for text segmentation is derived from Wikipedia article text with section boundaries. In Entity Aspect Linking, the goal is to refine an existing entity link in text to state which of several aspects of an entity is most relevant. Benchmarks were created from passages that contain hyperlinks to a Wikipedia article's section, where the article identifies the entity, and the section its most relevant aspect [20, 24].

In multi-hop question answering relevant information across multiple documents need to be combined to answer the question. Welbl et al. [27] construct the Quangaroo dataset from a chain of relation triples on Wikidata. For each entity in a relation chain, the lead texts of the entity's articles are provided. Previously, slot-filling and relation extraction approaches have been trained to extract information from Wikidata or Wikipedia infoboxes from the accompanying article [16, 19].

2.3 Related Resource Releases

As part of the TREC Complex Answer Retrieval (CAR) track[6], we provided a *Wikimark* based on a dump of the English Wikipedia from 2016. However, to avoid inadvertent leakage of test data leakage in the TREC CAR track, the software and toolchain to produce the *Wikimark* was not released (but we release it now).

Since the dump provided by Wikimedia is difficult to parse, we provided an easily-machine readable dump for English Wikipedia from 2018 and 2020 as a service to TREC News and TREC Conversational Assistance tracks.

Ramsdell et al. [24] provided a Wikimark for Entity Aspect Linking based on the 2020 English Wikipedia dump.

[5]https://fever.ai/dataset/feverous.html
[6]https://trec-car.cs.unh.edu

2.4 What is New about *this* Resource?

In contrast to previous releases, this resource provides software and customizable toolchain for converting Wikipedia dumps and harvesting *Wikimarks*. We are providing Wikipedia dumps for recent dumps from January 1st 2022. In addition to previously offered English Wikipedia dumps, we also provide conversions from Simple English and Japanese Wikipedias. We provide *Wikimarks* for query-specific clustering and entity linking, alongside previously available benchmarks for retrieval. Moreover, all datasets are now available as gzipped JSONL formats, although the previously used CBOR formats are offered as well.

3 RESOURCE CREATION APPROACH

Our resource is created by converting the Wikipedia dump to an easily machine-readable format, then processing it, extracting subsets from which *Wikimarks* are derived for a range of tasks.

3.1 Wikipedia Dump Conversion

MediaWiki offers "XML dumps" of the raw Wikitext markup,[7] as it was edited by authors of Wikipedia articles. A major challenge is the abundance of markup mistakes (due to the manual editing). The Wikitext markup is designed to be rendered by an error tolerant template engine that iteratively replaces markup with rendered text. There is no clear specification of how the markup is to be interpreted in the light of errors.[8]

A common source of errors arise from user mistakes in parenthetical markup expressions, such as links, templates, bold and italics markup or quotes which have missing end markers. Such mistakes renders Wikitext a very difficult format to parse. We use a PEG parser-generator that is able to efficiently backtrack and recover when content contains ill-formed expressions according to the grammar. However, we abandoned attempts of parsing italics and bold formatting, as these contain the highest numbers of markup errors (and hence parsing ambiguities).

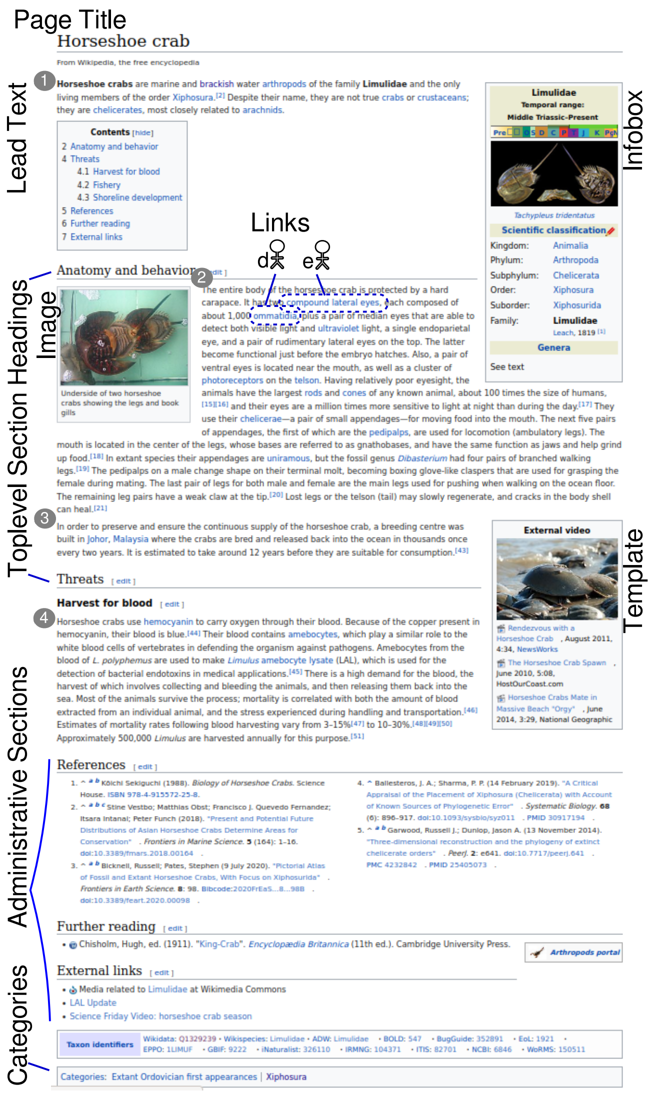
MediaWiki uses a variety of inline templates for formatting, tagging, common patterns, and temporal expressions. For example the templates `when` and `as-of` keep time expressions up to date. While by default, our processing pipeline deletes template expressions, our the language specific configuration file allows to provide static substitutions to avoid incomplete sentences.

Since we developed this parser in 2016, alternative Wikipedia access methods became available, such as MediaWiki's Parsoid.[9] While most of these libraries are targeting HTML and clear-text generation, our focus is on preserving all the semantic information associated with different elements of a Wikipedia article. In particular, we provide the following information about Wikipedia articles, as depicted in Figure 1:

Title: Used as page name and entity name. Page IDs are derived as URL-encoding of the page name. Example "Horseshoe crab".

Wikidata QID: The Wikidata IDs (`Qxxx`) is exposed as page metadata along with the wiki site identifier (e.g. "enwiki"). This allows cross-referencing corresponding pages across different snapshots and languages. Example "`Q1329239`".

[7]https://en.wikipedia.org/wiki/Help:Wikitext
[8]https://www.mediawiki.org/wiki/Markup_spec
[9]https://www.mediawiki.org/wiki/Parsoid



Horseshoe crab

From Wikipedia, the free encyclopedia

Horseshoe crabs are marine and brackish water arthropods of the family **Limulidae** and the only living members of the order Xiphosura.[2] Despite their name, they are not true crabs or crustaceans; they are chelicerates, most closely related to arachnids.

Contents [hide]
2 Anatomy and behavior
4 Threats
4.1 Harvest for blood
4.2 Fishery
4.3 Shoreline development
5 References
6 Further reading
7 External links

Limulidae
Temporal range: Middle Triassic–Present

Tachypleus tridentatus

Scientific classification

Kingdom:	Animalia
Phylum:	Arthropoda
Subphylum:	Chelicerata
Order:	Xiphosura
Suborder:	Xiphosurida
Family:	**Limulidae** Leach, 1819 [1]

Genera

See text

Anatomy and behavior [edit]

Underside of two horseshoe crabs showing the legs and book gills

The entire body of the horseshoe crab is protected by a hard carapace. It has two compound lateral eyes, each composed of about 1,000 ommatidia, plus a pair of median eyes that are able to detect both visible light and ultraviolet light, a single endoparietal eye, and a pair of rudimentary lateral eyes on the top. The latter become functional just before the embryo hatches. Also, a pair of ventral eyes is located near the mouth, as well as a cluster of photoreceptors on the telson. Having relatively poor eyesight, the animals have the largest rods and cones of any known animal, about 100 times the size of humans, [15][16] and their eyes are a million times more sensitive to light at night than during the day.[17] They use their chelicerae—a pair of small appendages—for moving food into the mouth. The next five pairs of appendages, the first of which are the pedipalps, are used for locomotion (ambulatory legs). The mouth is located in the center of the legs, whose bases are referred to as gnathobases, and have the same function as jaws and help grind up food.[18] In extant species their appendages are uniramous, but the fossil genus *Dibasterium* had four pairs of branched walking legs.[19] The pedipalps on a male change shape on their terminal molt, becoming boxing glove-like claspers that are used for grasping the female during mating. The last pair of legs for both male and female are the main legs used for pushing when walking on the ocean floor. The remaining leg pairs have a weak claw at the tip.[20] Lost legs or the telson (tail) may slowly regenerate, and cracks in the body shell can heal.[21]

In order to preserve and ensure the continuous supply of the horseshoe crab, a breeding centre was built in Johor, Malaysia where the crabs are bred and released back into the ocean in thousands once every two years. It is estimated to take around 12 years before they are suitable for consumption.[43]

External video

Rendezvous with a Horseshoe Crab , August 2011, 4:34, NewsWorks

The Horseshoe Crab Spawn , June 2010, 5:08, HostOurCoast.com

Horseshoe Crabs Mate in Massive Beach "Orgy" , June 2014, 3:29, National Geographic

Threats [edit]

Harvest for blood [edit]

Horseshoe crabs use hemocyanin to carry oxygen through their blood. Because of the copper present in hemocyanin, their blood is blue.[44] Their blood contains amebocytes, which play a similar role to the white blood cells of vertebrates in defending the organism against pathogens. Amebocytes from the blood of *L. polyphemus* are used to make *Limulus* amebocyte lysate (LAL), which is used for the detection of bacterial endotoxins in medical applications.[45] There is a high demand for the blood, the harvest of which involves collecting and bleeding the animals, and then releasing them back into the sea. Most of the animals survive the process; mortality is correlated with both the amount of blood extracted from an individual animal, and the stress experienced during handling and transportation.[46] Estimates of mortality rates following blood harvesting vary from 3–15%[47] to 10–30%.[48][49][50] Approximately 500,000 *Limulus* are harvested annually for this purpose.[51]

References [edit]

1. ^ a b Kōichi Sekiguchi (1988). *Biology of Horseshoe Crabs*. Science House. ISBN 978-4-915572-25-8.
2. ^ a b c Stine Vestbo; Matthias Obst; Francisco J. Quevedo Fernandez; Itsara Intanai; Peter Funch (2018). "Present and Potential Future Distributions of Asian Horseshoe Crabs Determine Areas for Conservation" . *Frontiers in Marine Science*. **5** (164): 1–16. doi:10.3389/fmars.2018.00164 .
3. ^ a b Bicknell, Russell; Pates, Stephen (9 July 2020). "Pictorial Atlas of Fossil and Extant Horseshoe Crabs, With Focus on Xiphosurida" . *Frontiers in Earth Science*. **8**: 98. Bibcode:2020FrEaS...8...98B . doi:10.3389/feart.2020.00098 .
4. ^ Ballesteros, J. A.; Sharma, P. P. (14 February 2019). "A Critical Appraisal of the Placement of Xiphosura (Chelicerata) with Account of Known Sources of Phylogenetic Error" . *Systematic Biology*. **68** (6): 896–917. doi:10.1093/sysbio/syz011 . PMID 30917194 .
5. ^ a b Garwood, Russell J.; Dunlop, Jason A. (13 November 2014). "Three-dimensional reconstruction and the phylogeny of extinct chelicerate orders" . *PeerJ*. **2**: e641. doi:10.7717/peerj.641 . PMC 4232842 . PMID 25405073 .

Further reading [edit]

- Chisholm, Hugh, ed. (1911). "King-Crab". *Encyclopædia Britannica* (11th ed.). Cambridge University Press.

Arthropods portal

External links [edit]

- Media related to Limulidae at Wikimedia Commons
- LAL Update
- Science Friday Video: horseshoe crab season

Taxon identifiers: Wikidata: Q1329239 • Wikispecies: Limulidae • ADW: Limulidae • BOLD: 547 • BugGuide: 352891 • EoL: 1921 • EPPO: 1LIMUF • GBIF: 9222 • iNaturalist: 326110 • IRMNG: 104371 • ITIS: 82701 • NCBI: 6846 • WoRMS: 150511

Categories: Extant Ordovician first appearances | Xiphosura

Figure 1: Example of semantic information provided by Wikipedia. https://en.wikipedia.org/wiki/Horseshoe_crab License: CC BY-SA 3.0; Redacted for brevity.

Paragraphs: Each paragraph is represented as chunks of plain text and links. The paragraph ID is derived from an MD5 hash of the visible text.

Links: Each internal hyperlink is preserved with anchor text, target page, and if applicable, target section. Example the text "brackish" links to the page "Brackish water".

Lead Text: Used as a short description of the entity, it is preserved as one or more paragraphs. Example: "Horseshoe crabs are marine…".

Sections and Headings: Sections are preserved with content and headings. Sections can contain subsections recursively. Heading IDs are derived as URL-encoding of the heading text. Example: "Threats" (top-level section).

Administrative Sections: Some sections include meta information about a page rather than actual content, often such sections are available for all Wikipedia pages, examples are "References", "Further reading", or "External links". Our pipeline will first convert them as-is but supports the removal of those sections if desired.

Lists: Bulleted lists are preserved with indentation level. The content is represented as a paragraph.

Images: Images are preserved with URL to the graphic and their caption, which is represented as a paragraph. Example "Underside of two horseshoe crabs…".

Infobox: Infoboxes are named (Example "biota") and represented as a list of key, value pairs. Keys are strings, while values can be any element above (paragraph, image, etc).

Categories: The category information is preserved in the page metadata (Example "Xiphosura").

Redirects: When a page is renamed, the old page name presents a redirect to the new page. This is a useful resource of alternative names. We post-process the dump to expose this information in the page metadata. Example: "Horseshoe crabs" (plural).

In-Links: While outlinks of a page can be derived from the link information of a page, we post-process all pages to collect hyperlinks linking to any given page. This information is preserved in the page metadata, both as page IDs and page names.

Disambiguations: Disambiguation pages that refer to this page as exposed in the page metadata. Disambiguation pages provide information about pages that share an ambiguous name. The Wikitext denotes this information as a disambiguation template. Unfortunately, different languages can use language-specific disambiguation templates for example in German "Begriffsklärung". These have to be customized in the language-specific pipeline configuration.

Page tags: Some interesting page information is preserved as page-level templates, examples are "Vital articles" or "Good articles" which are identified by a committee. The tags are configurable and will be exposed as page metadata.

The conversion results are available in the `unprocessedAll` package, which comprises article, category, and disambiguation pages. This package is most useful to analyze conversion errors or templates that need to be customized in the language-dependent configuration. Researchers who prefer to build their own *Wikimark* processing pipeline, are recommended to base their code on this data set.

3.2 Processing and Deduplication

The following pipeline steps will filter and deduplicate the dump before splitting it into *Wikimarks*.

Remove pages. In the next step, we discard category, disambiguation, and list pages since the information they provide is preserved in the metadata of article pages. The exact filter predicate can be configured by adjusting the `config.filterPredicates` property.

Remove sections. Then the content of each article is transformed by removing infoboxes, the category information at bottom of the page, and administrative headings. Page metadata is preserved. The filtering options can be configured with `config.pageProcessing`.

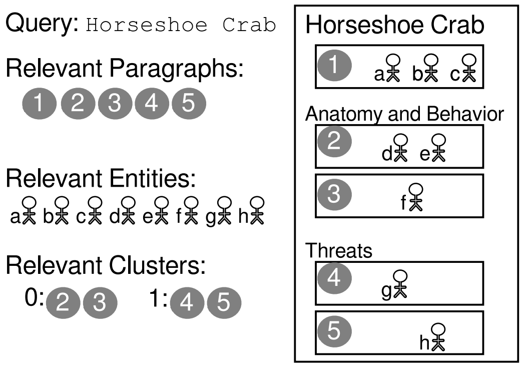


Figure 2: *Wikimarks* derived for article-level retrieval and clustering (left) from a given article (right). Paragraph IDs indicated by numbers in black dots; entity IDs as letters in stick figures; ground truth cluster index 1 identifies all paragraphs in the section "Threats".

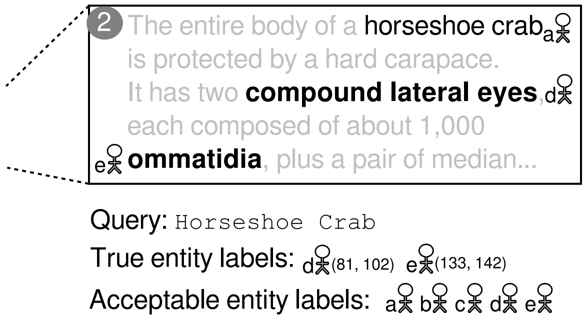

2 The entire body of a horseshoe crab a
is protected by a hard carapace.
It has two compound lateral eyes, d
each composed of about 1,000
e ommatidia, plus a pair of median...
Query: Horseshoe Crab
True entity labels: d(81, 102) e(133, 142)
Acceptable entity labels: a b c d e

Figure 3: *Wikimarks* derived for relevant entity linking (bottom) from a the second paragraph (top). The task is to annotate the plain text with entity links (for example with entities a, d, and e). True entities d and e are derived from hyperlinks contained in this paragraph (bold) with given character spans. Since entity a was linked in a previous paragraph and its annotation is to be accepted without penalty.

Deduplication (optional). There is an abundance of duplicated content on Wikipedia. Since the paragraph ID is based on a hash of the visible content, this provides an easy solution for removing identical content. However, there still remain many near-duplicates. For English pages, we provide a deduplication mechanism: All paragraphs are divided into buckets with a GloVE-based locality-sensitive hash. Next, every pair of paragraphs within the same bucket that has 95% bigram overlap is called a duplicate. After repeating the randomized process five times and applying the transitivity of duplicate relationships, we derive clusters with near-duplicate paragraphs. For each cluster, one representative paragraph is chosen.

Next all remaining articles in the collection are re-written to replace paragraphs in the duplicate cluster with its cluster representative. All following pipeline steps apply to this collection.

Paragraph Corpus. All paragraphs are extracted from of all articles, shuffled, and provided as a paragraph corpus.

Article Cleaning. To ensure that only high-quality text-centric articles are chosen during the *Wikimark* creation, we remove sections with very short headings (less than three letters) or very long headings (more than 100 characters) as these indicate pages with mal-formed Wikitext. We remove images and their captions as well as sections without textual content. We also remove any page with less than three remaining sections, as these indicate pages based on visual content or low-quality article stubs.

Wikimarks. We recommend to derive *Wikimarks* from cleaned articles in this set. The processing pipeline will associate each article with a train vs test split and one of five folds. To obtain random, yet reproducible results, we use a deterministic hash of the page name to assign a page to one of these splits, which are consistent across all following steps.

Our pipeline allows the selection of subsets of articles via a wide range of predicates described in Table 2. For example, subsets can be selected on the basis of category membership, existence of a page tag, or when explicitly listed via page title or Wikidata QID. Several predicates can be combined with simple logic expressions.

4 AUTOMATIC BENCHMARKS AKA *WIKIMARKS*

Wikimarks are created from a subset of pages, such as lists of Wikidata QIDs or category memberships. The page subset is separated into a test set and five train folds. For each of the task-specific datasets, such as queries, candidate sets, and relevance ground truths for the *Wikimarks* are exported. By default the following information is provided for each dataset:[10]

Articles †: Content of processed articles (JSONL or CBOR).
Titles/QIDs: Page titles and Wikidata QIDs of pages in this subset.
Paragraphs †: Corpus of paragraphs from this article subset.
Provenance: Information about the Wikipedia dump the subset originated from.

Additionally, task-specific *Wikimark* data is provides as described in the following.

4.1 Retrieval *Wikimark*

The retrieval *Wikimark* is designed to study the quality of retrieval models. For queries derived from Wikipedia titles, any paragraph originating from the Wikipedia article is counted as relevant. This *Wikimark* was referred to as the "automatic ground truth" in the TREC Complex Answer Retrieval task.

Wikimarks for three kinds of retrieval scenarios are provided:

- Article: The query is the page title, and the goal is to retrieve paragraphs that are relevant for this query. For the passage retrieval relevance data (i.e., qrels) any paragraph located anywhere on the original page is counted as relevant, all other paragraphs are non-relevant.

[10]Data marked with † is reserved for training and evaluation only.

- Toplevel: The query is a combination of page title and heading of a top-level section. The goal is to retrieve paragraphs that are in fact located within this section or one of its subsections.
- Hierarchical: The query is derived from any section on the page. The goal is to retrieve paragraphs that are exactly in this section, not a subsection.

In addition to passage-level retrieval, we also provide a *Wikimark* for entity retrieval, where any entity (as represented by their Wikipedia pages) that is linked to from a relevant paragraph is regarded as relevant.

As a corpus for retrieving passages from, we recommend to use the paragraph corpus described in Section 3.2. As a legal set of entities, we recommend to use an unprocessed dump of Wikipedia pages, such as `unprocessedAllButBenchmark`.

The retrieval *Wikimark* includes the following information:

Outlines: Title and section outlines of the articles, to derive query texts from. Page metadata is available.

Topics: Query IDs for each section—these can also be obtained from the outlines.

Passage Qrels †: `Trec-eval` compatible qrels files of paragraph IDs for article-level retrieval, top-level section retrieval, and hierarchical section retrieval.

Entity Qrels †: `Trec-eval` compatible qrels files of relevant entity IDs (same as page IDs) for article, top-level section, and hierarchical section retrieval.

Evaluation. We recommend to use the retrieval evaluation tool `trec-eval`[11] with option `-c` using the provided qrels files.

4.2 Query-specific Clustering *Wikimark*

The task of search result clustering, will, given a search query and a ranking of search results, identify query-specific clusters for presentation. We provide a *Wikimark* dataset for this clustering task, where the query is taken from the page title, and each top-level section defines one ground truth cluster. The search results are taken from the article-level retrieval task. To train on this task in isolation from a retrieval system, we suggest to use all passages that originate from this page.

The query-specific clustering *Wikimark* is provided as a JSONL gzipped file[12] which contains the following information:

Query: The query text is derived from the page name; the query ID from the page ID.

Elements: List of paragraph IDs contained on the page.

True Cluster Labels †: List of true cluster labels for each element. The i'th cluster label is derived from the section ID of the top-level section where the i'th element is located.

True Cluster Index †: Projecting the true cluster labels onto integers from $0, 1 \ldots$.

We remove instances with less than two clusters.

Evaluation. Our *Wikimark* format is designed to interface with `scikit.learn`'s cluster evaluation measures.[13] This evaluation assumes that true clusters are represented as indexes $0, 1, \ldots$. The evaluation measure uses a list of true labels, were for the `element` at position i, the true cluster index is located at position i of the list `true cluster label`. To evaluate a predicted clustering, represent each cluster on its own scale from $0, 1, \ldots$ (different from true labels), an produce a list of predicted cluster labels in the order of given elements i. We recommend to evaluate using the Adjusted RAND index evaluation metric, which ranges from -1 (worst) to +1 (best) with 0 referring to a random clustering.

[11]https://github.com/usnistgov/trec_eval
[12]Because of the redundancy induced by repeated JSON keys, we recommend to open the file as a Gzipped stream, rather than decompressing it.
[13]https://scikit-learn.org/stable/modules/clustering.html

Table 1: Number of articles in each *Wikimark* subset.

	en	simple	ja
vital-articles.test	521	461	503
vital-articles.train	528	471	539
good-articles.test	17,086	1	809
good-articles.train	17,361	2	838
US-history.test	4,232	9	--
US-history.train	4,284	13	--
horseshoe-crab.train	1	1	--
benchmarkY1.test	131	44	71
benchmarkY1.train	117	42	81
car-train-large.train	884,709	17,335	246,649
test200.test	--	1	42
test200.train	188	12	44

4.3 Relevant Entity Linking *Wikimark*

Entity linking is typically discussed as an NLP task that ignores the context of a search query. However when presenting relevant information for a search query, maybe it would be best not to annotate all possible entity links, but instead focus on linking entities that are relevant for the query. *Wikimarks* allow us to create a query-oriented entity linking dataset, as Wikipedia's editorial policies are to only include hyperlink to pages when the information is relevant for the topic of the article.

The relevant entity linking *Wikimark* is provided as a JSONL gzipped file which contains the following information:

Query: The query text is derived from the page name; the query ID from the page ID.

Text-only Paragraph: The text contents of paragraph (without entity links), to be annotated with entity links.

True Linked Paragraph †: The original paragraph (with links) for training and as ground truth.

True Entity Labels †: List of entity IDs that should be linked in this paragraph. These are provided as internal page IDs as well as Wikidata QIDs.

Acceptable Entity Labels †: List of acceptable entity IDs that can be linked in this paragraph without penalty. List of entities linked in this paragraph and any previous paragraph. These are provided as internal page IDs as well as Wikidata QIDs.

We remove instances of paragraphs without any linked entities. Wikipedia's editorial policies mandate that entities are only linked once per article. Consequently, entities that are mentioned repeatedly are only linked once. Since the entity linking ground truth is derived from hyperlinks, entity linking predictions would get penalized for linking all these entities. To alleviate this without resorting to heuristics, we collect all entities linked in all preceding paragraphs of an article and exposed them as `acceptable entity labels`. The entity linking evaluation should only give credit to entities in `true labels`, but not penalize entities in `acceptable labels`.

Evaluation. Unfortunately, there are no established evaluation frameworks for entity linking. We recommend to separately evaluate the set of correctly linked entities and the offsets of annotated spans as macro-average on the mean-squared error of character offsets. A widely used evaluation measure is the F1 measure on the predicted entity set (referred to as `predicted labels`). For the F1 measure we suggest to derive statistics from the size of label sets as follows.

TP: `predicted labels` ∩ `true labels`
FP: `predicted labels` \ `acceptable labels`
FN: `true labels` \ `predicted labels`

We suggest to evaluate the quality of annotated text spans via the set of all true labels T. The per-entity span error is the root mean squared error on predicted spans (b_p, e_p) versus spans (b_t, e_t).

Interfacing with Entity Linking Software. A wide range of entity linking software is available, such as the TagMe-successor "WAT"[14] or Rel [26]. A practical issue lies in using unique identifiers of entities in linking systems. While more and more entity linking systems are using stable Wikidata QIDs, today, many entity linking systems still use English Wikipedia page titles to uniquely identify entities—even though pages are occasionally renamed, and page titles on Simple English might be different.

While the Wikidata API provides a lookup-service, it is not practical for large-scale entity linking experiments. To support this situation, we provide a lookup resource for converting Wikipedia page titles to Wikidata QIDs, which includes previously renamed titles.

5 PROVIDED DUMPS AND *WIKIMARKS*

Since running our conversion pipeline can take several days, we demonstrate its usefulness by providing three Wikipedia dumps from January 1st, 2022 for English Wikipedia, Simple English Wikipedia, and Japanese Wikipedia.

For each Wikipedia, we provide the following information both as CBOR and JSONL output formats:

`unprocessedAll`: an unprocessed dump of all Wikipedia articles, as described in Section 3.1.
`collection`: all resources needed to perform experiments:
- **`benchmarks`:** *Wikimarks* for different subsets, as detailed below.
- **`paragraphCorpus`:** a corpus for passage retrieval as described in Section 3.2.
- **`unprocessedAllButBenchmark`:** an unfiltered and unprocessed Wikipedia dump in lieu of a knowledge graph, pages from which *Wikimarks* were created are held out.

We provide an example *Wikimarks* to demonstrate the versatility of the pipeline. These subsets are highly configurable through the filter predicates (examples below) and can be extended in the Wikipedia-specific configuration. Table 1 lists the number of pages contained in each train/test set of the subsets.

Vital-articles: A set of important articles that the Wikipedia community identified.[15] The community strives to provide these articles for all languages. We obtain the set of vital articles via Wikidata, then filter the processed articles by Wikidata QID. Predicate `qid-set-from-file "./vital-articles.qids"`.
Good-articles: A Wikipedia committee defines a set of good articles[16] that are well-written, contain factually accurate and verifiable information and are of broad importance. Such pages are identified either as template "GA" or "good article", which our pipeline is configured to expose as page tag "Good article". Predicate: `has-page-tag ["Good article"]`.
US-history: A set of pages in categories that contain the words "United" "States" "history", such as "History of the United States" or "United States history timelines". Predicate: `(category-contains "history" & category-contains "united" & category-contains "states")`.
Horseshoe-crab: The single Wikipedia page on horseshoe crabs used in the example above. It is identified by its Wikidata QID. Predicate: `qid-in-set ["Q1329239"]`.

For backwards compatibility, we also provide subsets used in the TREC Complex Answer Retrieval track.

BenchmarkY1: Train/test set used in the first year of TREC CAR Articles manually selected by browsing then randomly split into train/test.
Car-train-large: Intended to be a very large training set for CAR, excludes pages with categories related to people, events, works of art, sports clubs and lists. This set was originally known in CAR as "train-v2.0" and used to filter the training set. Note, that the general training set produced by our pipeline includes all categories and different filter criteria can be configured.

Table 2: Predicate syntax for selecting subsets of pages. Several variants also allow to provide IDs as a file.

name-contains SUBSTR	substring match in page names
name-has-prefix PREFIX	prefix match in page names
name-has-suffix SUFFIX	suffix match in page names
category-contains SUBSTR	substring match in page's category
name-in-set ["P1", "P2", "P3"]	exact pagename match in set P1,P2,P3
name-or-redirect-in-set ["P1", ..]	same as name-in-set but also matches in redirects (useful when pages are renamed)
pageid-in-set ["P1", "P2", "P3"]	exact page ID match in set
qid-in-set ["Q1", "Q2", "Q3"]	exact Wikidata QID match in set
has-page-tag ["T1", ...]	page tag match, e.g. "Good article"
PRED1 | PRED2	Boolean OR
PRED1 & PRED2	Boolean AND
! PRED	Boolean NOT
page-hash-mod N K [SALT]	create page hash for train, test, and fold split: hash of the page name mod $N == K$.

[14]https://sobigdata.d4science.org/web/tagme/wat-api

[15]Vital articles https://en.wikipedia.org/wiki/Wikipedia:Vital_articles
[16]Good articles https://en.wikipedia.org/wiki/Wikipedia:Good_articles

Table 3: Number of relevant passages, relevant entities, cluster instances, and entity linking instances in train/test article-level instances derived from each subset. Each article produces one query, see Table 1 for number of articles in subssets.

	Relevant Passages			Relevant Entities			Clustering Instances			Entity Linking Instances		
	en	simple	ja	en	simple	ja	en	simple	ja	en	simple	jp
vital-articles.test	44,444	7,117	12,448	159,392	20,626	38,975	521	328	393	64,857	9,339	23,217
vital-articles.train	42,008	6,845	13,330	149,609	19,401	42,357	528	324	440	61,984	8,663	25,743
good-articles.test	408,454	7	23,869	1429,087	47	65,031	17,088	1	626	777,081	8	27,903
good-articles.train	415,034	17	24,375	1465,327	87	61,050	17,362	1	626	789,726	39	27,538
US-history.test	83,213	176	–	206,672	405	–	4,232	6	–	169,014	210	–
US-history.train	83,255	146	–	205,438	608	–	4,285	7	–	160,764	173	–
horseshoe-crab.train	21	11	–	69	40	–	1	1	–	44	13	–
benchmarkY1.test	6,554	434	1,160	15,698	1,117	3,018	131	23	56	8,536	454	1,978
benchmarkY1.train	5,588	449	1,396	14,744	1,273	3,440	117	25	60	7,258	513	2,152
car-train-large.train	9,254,925	113,444	1496,289	19,764,159	249,369	3,462,123	885,014	6,918	87,012	25,423,934	185,203	3824,333
test200.train	5,537	109	335	12,345	272	929	188	5	19	9,147	135	612

Test200: An initial test of two hundred training pages selected from a list of articles in training fold 0. Some of these articles were later identified to be of weak quality (and removed from Wikipedia) and this subset was not officially used.

Since `car-train-large` and `test200` subsets do not provide a test set, all files in the test directory are empty.

6 EXPERIMENTAL REFERENCE BASELINES

We are providing reference baselines for our example *Wikimarks* derived for the benchmarkY1 subsets from English and Simple English Wikipedia. The total number of test instances for each *Wikimark* are given in Table 3. The results obtained with our recommended evaluation approaches are presented in Table 4.

Continuously updated reference results are available on the online appendix.[17]

6.1 Passage Retrieval

Reference results are provided for article-level retrieval, where page titles are used as queries. Paragraphs are retrieved from the `paragraphCorpus` of each Wikipedia dump respectively.

Baseline implementations are based on Lucene, with code provided online.[18] As baselines we include:

bm25: Lucene's BM25 method.
bm25-rm3: RM3 query expansion, then retrieve with BM25.
QL-rm3: RM3 query expansion, then retrieve with Lucene's Dirichlet-smoothed query likelihood.

As can be seen in Table 4, all methods perform reasonably in terms of MAP. For brevity we are omitting R-precision, reciprocal rank, and NDCG results, which lead to similar conclusions.

Note that in this study, article-level results are reported—this is unlike results for TREC CAR which are based on hierarchical-level retrieval.

6.2 Entity Retrieval

In article-level entity retrieval, page titles are used as queries. Entities are retrieved via their corresponding pages of each respective Wikipedia dump.

Baseline implementations are based on Lucene using a search index of pages in `allButBenchmark` and an index of paragraphs (with links) of the `paragraphCorpus`. As baselines we include:

page-bm25: Retrieving Wikipedia pages via BM25.
page-bm25-rm3: RM3 query expansion, then retrieving pages with BM25.
paragraph-bm25-ECM: Retrieving paragraphs with BM25, then ranking entities linked in these paragraphs with the entity context model (ECM).

The entity context model (ECM) [8] represents documents in a feedback run as a language model over entity links, then uses an RM3-style expansion to derive a distribution over entity links. While Dalton et al. [11] used these entities as expansion terms, here we use them as a means of ranking entities.

Table 4 demonstrates that retrieving entities via their Wikipedia pages obtains poor results, while ECM, which uses entity links in a feedback set, obtains much better results. This is in line with earlier findings [8].

6.3 Clustering

We provide reference results for the following clustering methods based on TF-IDF and clustering in default configuration as provided by `scikit.learn`.[19] Here we assume knowledge of the true number of clusters, but cluster methods that predict the number of clusters can also be evaluated with this benchmark.

TF-IDF agglomerative: Each paragraph is represented as a TF-IDF vector, then using agglomerative clustering with Euclidean distance.
TF-IDF kmeans: TF-IDF paragraph representation, then using K-means clustering.
SBERT kmeans: Using Sentence-BERT paragraph representation, then using K-means clustering.

Sentence-BERT [25] is a BERT-based embedding model trained for clustering sentences. We are using the `bert-base-uncased` version provided by the authors.

We evaluate the predicted clusterings using the Adjusted Rand Index. Results are given in Table 4. We observe that Sentence-BERT performs best.

[17] https://trema-unh.github.io/wikimarks/evaluation.html   Please send your results!
[18] Passage and Entity Retrieval: https://github.com/laura-dietz/trec-car-methods

[19] `sklearn.feature_extraction.text` and `sklearn.cluster` in version 1.0.2

Table 4: Results obtained by baselines across train/test subsets of benchmarkY1 derived from both English and Simple English Wikipedias. Average performance is provided with standard-error bars (rounding non-significant digits). Best baseline for each task marked in bold. Paragraph and entity retrieval are evaluated by trec_eval in Mean-average precision. Query-specific clustering is evaluated by scikit.learn in Adjusted RAND Index. Entity linking is evaluated with the F1-measure, macro-averaged across paragraphs. Additional reference results are available in the online appendix.

	simple		en	
	benchmarkY1.train	benchmarkY1.test	benchmarkY1.train	benchmarkY1.test
Paragraph Retrieval [MAP]				
bm25	**0.31 ± 0.04**	**0.29 ± 0.03**	0.097 ± 0.01	0.094 ± 0.01
bm25-rm3	0.29 ± 0.04	0.26 ± 0.03	**0.107 ± 0.01**	**0.101 ± 0.01**
QL-rm3	0.25 ± 0.04	0.20 ± 0.02	0.084 ± 0.01	0.076 ± 0.01
Entity Ranking [MAP]				
page-bm25	0.03 ± 0.005	0.038 ± 0.007	0.025 ± 0.002	0.026 ± 0.003
page-bm25-rm3	0.05 ± 0.007	0.048 ± 0.007	0.037 ± 0.003	0.038 ± 0.004
paragraph-bm25-ECM	**0.23 ± 0.03**	**0.253 ± 0.021**	**0.215 ± 0.01**	**0.21 ± 0.01**
Cluster [Adjusted RAND Index]				
TF-IDF agglomerative	0.16 ± 0.06	0.27 ± 0.07	0.15 ± 0.01	0.16 ± 0.01
TF-IDF kmeans	0.13 ± 0.01	0.12 ± 0.01	0.11 ± 0.04	**0.19 ± 0.05**
SBERT kmeans	**0.38 ± 0.09**	**0.38 ± 0.09**	**0.23 ± 0.02**	**0.19 ± 0.01**
Entity Linking [Paragraph-Macro-avg F1]				
WAT	**0.44 ± 0.01**	**0.42 ± 0.01**	**0.332 ± 0.004**	**0.310 ± 0.003**

6.4 Entity Linking

We provide query-agnostic entity linking reference results for with the WAT entity linker[20] [23] using its default configuration. Note that this approach is query-agnostic, further research is required to study the utility of incorporating the query into the entity link prediction.

Since the WAT linker used entity IDs based on Wikipedia titles of an unknown dump, we use our resource for converting (potentially renamed) titles to Wikidata QID to annotate entity linking spans with QIDs. We follow the evaluation paradigm discussed in Section 4.3. Results are evaluated in F1 per paragraph, we provide macro-averages over paragraphs.

We find that an external entity linking tool can obtain reasonable performance on our *Wikimark*. While results are omitted, we obtain similar performance for micro-averaged F1 (0.29), and query-based macro-averaged-F1 (0.30) for benchmarkY1.test set from English Wikipedia. We find 15,561 true positives, 70,120 false positives, and 4,965 false negatives.

7 CONCLUSIONS

We provide a resource for generating *Wikimarks*—automatically harvested benchmarks from Wikipedia. We provide an efficient implementation for deriving *Wikimarks* from any Wikipedia dump. Our pipeline handles the download of Wikipedia files, the dump conversion, processing, subset creation, and derivation of *Wikimarks*. We provide four example *Wikimarks* for passage and entity retrieval, query-specific clustering, and query-specific entity linking—but *Wikimarks* for other tasks can be created also. The pipeline is configurable to use different page subsets, different languages, and different filter criteria. While the software was originally developed to provide data for the TREC Complex Answer Retrieval track, it has been significantly extended towards language support, integration of Wikidata, and new JSONL output formats.

As a proof of concept, we provide three recent Wikipedia dumps for English, Simple English, and Japanese from January 1st, 2022, along with *Wikimarks* for page subsets. We provide results of reference baseline results for retrieval, clustering, and entity linking on these datasets, which are in-line with previous findings.

Other communities have successfully leveraged content from Wikipedias for benchmark creation, and studies on TREC CAR demonstrate that findings under automatic *Wikimarks* often agree with findings under manually created benchmarks. Nevertheless, *Wikimarks* are not a replacement for manually annotated benchmarks. However, we hope to contribute a means for providing evaluations for early research developments and training of data-hungry machine learning methods. This approach can overcome challenges for novel information retrieval tasks, such as search result organization and result annotation for information-seeking.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Science FoundationNational Science Foundation under Grant No. 1846017. Disclaimer. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

We thank Hideo Joho for providing the Japanese language configuration for this pipeline. We thank all participants TREC CAR for being early adopters, and anyone who engaged with us in discussion on how to derive additional benchmarks from Wikipedia.

[20]https://sobigdata.d4science.org/web/tagme/wat-api

REFERENCES

[1] Rami Aly, Zhijiang Guo, Michael Sejr Schlichtkrull, James Thorne, Andreas Vlachos, Christos Christodoulopoulos, Oana Cocarascu, and Arpit Mittal. 2021. FEVEROUS: Fact Extraction and VERification Over Unstructured and Structured information. arXiv:2106.05707 [cs.CL]

[2] Sebastian Arnold, Rudolf Schneider, Philippe Cudré-Mauroux, Felix A Gers, and Alexander Löser. 2019. SECTOR: A Neural Model for Coherent Topic Segmentation and Classification. *Transactions of the Association for Computational Linguistics* 7 (2019), 169–184.

[3] Nima Asadi, Donald Metzler, Tamer Elsayed, and Jimmy Lin. 2011. Pseudo test collections for learning web search ranking functions. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, 1073–1082.

[4] Leif Azzopardi, Maarten De Rijke, and Krisztian Balog. 2007. Building simulated queries for known-item topics: an analysis using six european languages. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 455–462.

[5] Steven M Beitzel, Eric C Jensen, Abdur Chowdhury, and David Grossman. 2003. Using titles and category names from editor-driven taxonomies for automatic evaluation. In *Proceedings of the twelfth international conference on Information and knowledge management*. ACM, 17–23.

[6] Richard Berendsen, Manos Tsagkias, Maarten De Rijke, and Edgar Meij. 2012. Generating pseudo test collections for learning to rank scientific articles. In *International Conference of the Cross-Language Evaluation Forum for European Languages*. Springer, 42–53.

[7] Richard Berendsen, Manos Tsagkias, Wouter Weerkamp, and Maarten De Rijke. 2013. Pseudo test collections for training and tuning microblog rankers. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 53–62.

[8] Shubham Chatterjee and Laura Dietz. 2021. Entity Retrieval Using Fine-Grained Entity Aspects. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1662–1666.

[9] Gordon V Cormack, Christopher R Palmer, and Charles LA Clarke. 1998. Efficient construction of large test collections. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. 282–289.

[10] Silviu Cucerzan. 2007. Large-scale named entity disambiguation based on Wikipedia data. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*. 708–716.

[11] Jeffrey Dalton, Laura Dietz, and James Allan. 2014. Entity Query Feature Expansion Using Knowledge Base Links. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval* (Gold Coast, Queensland, Australia) *(SIGIR '14)*. Association for Computing Machinery, New York, NY, USA, 365–374. https://doi.org/10.1145/2600428.2609628

[12] Bhavana Dalvi, Einat Minkov, Partha P Talukdar, and William W Cohen. 2015. Automatic gloss finding for a knowledge base using ontological constraints. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*. ACM, 369–378.

[13] Laura Dietz and Jeff Dalton. 2020. Humans optional? automatic large-scale test collections for entity, passage, and entity-passage retrieval. *Datenbank-Spektrum* 20, 1 (2020), 17–28.

[14] Demian Gholipour Ghalandari, Chris Hokamp, John Glover, Georgiana Ifrim, et al. 2020. A Large-Scale Multi-Document Summarization Dataset from the Wikipedia Current Events Portal. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 1302–1308.

[15] Hiroaki Hayashi, Prashant Budania, Peng Wang, Chris Ackerson, Raj Neervannan, and Graham Neubig. 2021. WikiAsp: A Dataset for Multi-domain Aspect-based Summarization. *Transactions of the Association for Computational Linguistics* 9 (2021), 211–225.

[16] Daniel Hewlett, Alexandre Lacoste, Llion Jones, Illia Polosukhin, Andrew Fandrianto, Jay Han, Matthew Kelcey, and David Berthelot. 2016. Wikireading: A novel large-scale language understanding task over wikipedia. *arXiv preprint arXiv:1608.03542* (2016).

[17] Gaya K Jayasinghe, William Webber, Mark Sanderson, and J Shane Culpepper. 2014. Improving test collection pools with machine learning. In *Proceedings of the 2014 Australasian Document Computing Symposium*. 2.

[18] Sumanta Kashyapi and Laura Dietz. 2022. Query-specific Subtopic Clustering. In *The annual Joint Conference on Digital Libraries (JCDL)*.

[19] Gjergji Kasneci, Maya Ramanath, Fabian Suchanek, and Gerhard Weikum. 2009. The YAGO-NAGA approach to knowledge discovery. *ACM SIGMOD Record* 37, 4 (2009), 41–47.

[20] Federico Nanni, Simone Paolo Ponzetto, and Laura Dietz. 2018. Entity-aspect linking: providing fine-grained semantics of entities in context. In *Proceedings of the 18th ACM/IEEE on Joint Conference on Digital Libraries*. 49–58.

[21] Joel Nothman, James R Curran, and Tara Murphy. 2008. Transforming Wikipedia into named entity training data. In *Proceedings of the Australasian Language Technology Association Workshop 2008*. 124–132.

[22] Laura Perez-Beltrachini and Mirella Lapata. 2021. Models and Datasets for Cross-Lingual Summarisation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 9408–9423.

[23] Francesco Piccinno and Paolo Ferragina. 2014. From TagME to WAT: a new entity annotator. In *Proceedings of the first international workshop on Entity recognition & disambiguation*. 55–62.

[24] Jordan Ramsdell and Laura Dietz. 2020. A Large Test Collection for Entity Aspect Linking. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 3109–3116.

[25] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 3982–3992.

[26] Johannes M van Hulst, Faegheh Hasibi, Koen Dercksen, Krisztian Balog, and Arjen P de Vries. 2020. Rel: An entity linker standing on the shoulders of giants. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2197–2200.

[27] Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. *Transactions of the Association for Computational Linguistics* 6 (2018), 287–302.

[28] Emine Yilmaz, Evangelos Kanoulas, and Javed A Aslam. 2008. A simple and efficient sampling method for estimating AP and NDCG. In *Proceedings of the SIGIR*. 603–610.

[29] Haotian Zhang, Gordon V Cormack, Maura R Grossman, and Mark D Smucker. 2018. Evaluating Sentence-Level Relevance Feedback for High-Recall Information Retrieval. *arXiv preprint arXiv:1803.08988* (2018).