

WikiBrain: Democratizing computation on Wikipedia

Shilad Sen
Macalester College
St. Paul, Minnesota
ssen@macalester.edu

Toby Jia-Jun Li
University of Minnesota
Minneapolis, Minnesota
lix2211@umn.edu

WikiBrain Team^{*}
Matt Lesicko, Ari Weiland, Rebecca Gold,
Yulun Li, Benjamin Hillmann
Macalester College
St. Paul, Minnesota

Brent Hecht
University of Minnesota
Minneapolis, Minnesota
bhecht@cs.umn.edu

ABSTRACT

Wikipedia is known for serving humans' informational needs. Over the past decade, the encyclopedic knowledge encoded in Wikipedia has also powerfully served computer systems. Leading algorithms in artificial intelligence, natural language processing, data mining, geographic information science, and many other fields analyze the text and structure of articles to build computational models of the world.

Many software packages extract knowledge from Wikipedia. However, existing tools either (1) provide Wikipedia data, but not well-known Wikipedia-based algorithms or (2) narrowly focus on one such algorithm.

This paper presents the WikiBrain software framework, an extensible Java-based platform that democratizes access to a range of Wikipedia-based algorithms and technologies. WikiBrain provides simple access to the diverse Wikipedia data needed for semantic algorithms and technologies, ranging from page views to Wikidata. In a few lines of code, a developer can use WikiBrain to access Wikipedia data and state-of-the-art algorithms. WikiBrain also enables researchers to extend Wikipedia-based algorithms and evaluate their extensions. WikiBrain promotes a new vision of the Wikipedia software ecosystem: every researcher and developer should have access to state-of-the-art Wikipedia-based technologies.

1. INTRODUCTION

In 2004, Jimmy Wales articulated his audacious vision for Wikipedia: "Imagine a world in which every single person on the planet is given free access to the sum of all human knowledge [31]." In the decade since Wales' declaration, Wikipedia

^{*}mmlesicko@gmail.com, math1man@gmail.com, rebecca-gold0@gmail.com, yulun.li2@gmail.com, bhillmann@outlook.com

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.

OpenSym '14, August 27 - 29 2014, Berlin, Germany. Copyright is held by the owner/author(s). Publication rights licensed to ACM. ACM 978-1-4503-3016-9/14/08 \$15.00. <http://dx.doi.org/10.1145/2641580.2641615>

has raced towards this goal, growing in articles, revisions, and gigabytes by factor of 20.¹ The planet has taken notice. Humans turn to Wikipedia's 31 million articles more than any other online reference. The online encyclopedia's 287 languages attract a global audience that make it the sixth most visited site on the web.²

However, outside the public eye, computers have also come to rely on Wikipedia. Algorithms across many fields extract meaning from the encyclopedia by analyzing articles' text, links, titles, and category structure. Search engines such as Google and Bing respond to user queries by extracting structured knowledge from Wikipedia.³ Leading semantic relatedness (SR) algorithms estimate the strength of association between words by mining Wikipedia's text, link structure [32], and concept structure [5]. Geospatial technologies extract geographic information from Wikipedia to understand the world around us in entirely new ways (e.g. [22, 20]).

Though powerful, these Wikipedia-centric algorithms present serious engineering obstacles. Downloading, parsing, and saving hundreds of gigabytes of database content requires careful software engineering. Extracting structure from Wikipedia requires computers to reconcile the messy, human process of article creation and revision. Finally, determining exactly what information an application needs, whether Wikipedia provides that information, and how that information can be efficiently delivered taxes software designers.

These engineering challenges stand as a barrier to developers of intelligent applications and researchers of Wikipedia-based algorithms. While large software companies such as Google and Microsoft extensively use Wikipedia-mined knowledge¹, smaller companies with limited engineering resources have a much harder time leveraging it. Researchers mining Wikipedia face similar engineering challenges. Instead of focusing on innovative contributions, they frequently spend hours reinventing the Wikipedia data processing wheel or wrangling a complex Wikipedia analysis framework to meet their needs. This situation has created a fragmented Wikipedia software ecosystem in which it is extremely difficult to reproduce results and extend Wikipedia-based technologies.

¹http://en.wikipedia.org/wiki/Wikipedia:Size_of_Wikipedia

²<http://www.alexa.com/topsites>

³<http://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html>

This paper introduces WikiBrain,⁴ a software library that democratizes access to state-of-the-art Wikipedia-based algorithms and techniques. In a few minutes, a developer can add the WikiBrain Java library and run a single program that downloads, parses, and saves Wikipedia data on commodity hardware. WikiBrain provides a robust, fast API to access and query article text and structured metadata such as links, categories, pageview analytics, and Wikidata.[49]. WikiBrain also makes three state-of-the-art research advances available to every developer and researcher as API “primitives:” a multilingual concept network (Section 4), semantic relatedness algorithms (Section 5), and geospatial data integration (Section 6). In doing so, WikiBrain empowers developers to create rich intelligent applications and enables researchers to focus their efforts on creating and evaluating robust, innovative, reproducible software.

This paper describes WikiBrain, compares it to related software libraries, and provides examples of its use. Section two begins by reviewing research algorithms that mine Wikipedia and software APIs that support those algorithms. Section three discusses the overall design and usage of WikiBrain. We then delve into details on the three ‘primitives’ (Sections four, five, and six). Finally, section seven consists of a case study that uses WikiBrain to quantitatively evaluate Tobler’s First Law of Geography – “everything is related to everything else, but near things are more related than distant things” [48] – replicating the results of a study by Hecht and Moxley [20] in a few lines of code.

2. RELATED WORK

Wikipedia has attracted the attention of researchers in a wide range of fields. Broadly speaking, Wikipedia research can be split into two categories: work that *studies* Wikipedia and work that *uses* Wikipedia as a source of world knowledge. While the former body of literature is large (e.g. [23, 41, 25, 34]), the latter is likely at least an order of magnitude larger.

Researchers have leveraged the unprecedented structured repository of world knowledge in Wikipedia for an enormous variety of tasks. For instance, Wikipedia has helped computers read natural language (e.g. [32, 7, 4]), it is slowly becoming the “crystallization point” for the semantic web [1, 39] (especially with the onset of Wikidata), and it has even been used to predict the future in domains like public health [28] and box office performance [29].

Due to the value of the encyclopedic world knowledge in Wikipedia, many software packages have been developed to extract information from Wikipedia and many datasets of extracted information have been shared. While these software packages and datasets are successful in a number of ways, they are either limited to a subset of basic data structures or focus on a single intelligent Wikipedia-based algorithm or system.

WikiBrain, in contrast, supports diverse Wikipedia technologies (e.g. spatial, semantic relatedness, and multilingual), and provides simple access to a variety of basic structures that have been shown to be useful in other technologies. With these structures (and WikiBrain’s algorithms), we hope to facilitate the incorporation of existing Wikipedia-based technologies and the development of new ones.

Below, we discuss some of the more prominent existing

software packages and datasets for processing Wikipedia data. We divide this discussion into two parts: the first is targeted at software and datasets related to basic Wikipedia structures and the second is focused on software that is specific to a single Wikipedia-based technology. As the WikiBrain team has the strict policy of not reinventing the wheel whenever possible, WikiBrain leverages some of this existing work in select portions of its functionality. Where this is the case, we describe how WikiBrain was aided by a particular software package or dataset.

2.1 Access to basic data structures

- **Wikipedia API:** The Wikimedia Foundation provides access to an impressive array of data through a REST API. Accessing this API is substantially slower than accessing well-engineered locally stored versions of the data. However, the Wikipedia API has one important advantage over all other means of accessing Wikipedia data: it is always 100% up to date. In order to support real-time access to Wikipedia data, WikiBrain has wrappers for the Wikipedia API that allow developers to access the live Wikipedia API just as easily as they can locally stored Wikipedia data (the switch is a single line of code per data structure). They can even easily integrate data from the two sources.
- **Java Wikipedia Library (JWPL):** JWPL provides access to basic data structures like links and categories. WikiBrain uses some libraries from JWPL to facilitate the parsing of wiki markup.
- **DBpedia:** DBpedia [1] is a highly influential project that extracts structured information from Wikipedia pages. Its widely-used datasets include basic structures like links and categories, but also an inferred general-purpose ontology, mappings to other data sources (e.g. Freebase, YAGO), and other enriched corpora. WikiBrain connects to DBpedia’s dataset of geographic coordinates as described below.
- **bliki:** Bliki is a “parser library for converting Wikipedia wikitext notation to HTML”. Bliki allows for structured access to a variety of wiki markup data structures (e.g. images, links, tables). However, Bliki does not support information external to wiki markup, for instance page views and Wikidata.
- **Python tools:** A number of Python tools exist to support access to basic Wikipedia data structures (e.g. Halfaker’s recently released MediaWiki Utilities [12]), but these target basic data.

2.2 Specialized intelligent tools

- **Wikipedia Miner:** Wikipedia Miner is a software library built around Milne and Witten’s link-based semantic relatedness measure [32], which we have also implemented in WikiBrain (with some extensions). While Wikipedia Miner provides fast and robust support for this measure and related services (e.g. disambiguation), it is specifically focused on this measure and does not include support or frameworks for additional semantic relatedness measures or other intelligent Wikipedia-based technologies (e.g. spatial and multilingual).
- **Semantic similarity:** A variety of software packages implement semantic similarity algorithms based on WordNet, a family of algorithms that led to current Wikipedia-based semantic relatedness algorithms (semantic similarity is a

⁴<http://wikibrainapi.org>

subset of semantic relatedness). SML-Library [14] and the Wordnet::Similarity Perl Library [37] are two of the most-known packages. Other software libraries in this area also exist, but they do not include the diversity of SR algorithms, performance optimizations, and flexibility of WikiBrain.

3. OVERVIEW OF WIKIBRAIN

This section provides an overview of the WikiBrain framework, explaining how users incorporate it into their projects and detailing its overall design and feature set. Specific Java code examples appear in Section 7.

3.1 Features and basic use

WikiBrain provides a rich set of intelligent “primitives” and a wide range of data structures, offering a more robust platform that related software packages. Below, we summarize a subset of these features:

- **Core data structures:** WikiBrain downloads and organizes Wikipedia datasets published by the Wikimedia foundation. It provides access to raw wikitext and article plaintext, including full-text search using Lucene.⁵ It also stores structured article metadata, including wikilinks, the redirect graph, and the category graph.
- **Multilingual support:** WikiBrain supports all Wikipedia language editions. It also includes algorithms that connect language-specific articles to their corresponding language-neutral concepts (Section 4).
- **Semantic relatedness algorithms.** WikiBrain includes six state-of-the-art semantic relatedness (SR) algorithms that measure the relationship between two concepts. As discussed below (Section 5), SR algorithms serve as a key component for a litany of intelligent technologies. WikiBrain also provides a robust set of evaluation tools for SR researchers.
- **Wikidata:** WikiBrain stores and queries structured Wikidata “facts” about articles. For example, birth dates of people, national languages of countries, and scientific taxonomies of animals.
- **Spatial infrastructure:** Wikibrain supports the storing and querying of geographic information connected to articles (Section 6).
- **Usability:** WikiBrain is fast and easy to use. It offers a variety of flexible configuration options and simple environment setup. WikiBrain supports optimized data representations for critical data structures such as links and single-machine parallelization (i.e. multi-threading support) for all computationally intensive features.
- **Pageviews** WikiBrain imports and analyzes page view logs that capture how many times each page is visited.⁶

Java projects incorporate WikiBrain by downloading the Wikibrain jar files or referencing the maven build dependency (see Section 7). Developers interact with WikiBrain in two stages. In the **import stage**, Wikipedia database dumps are downloaded, parsed, and stored. This pre-processing step occurs once. In the **runtime stage**, an application uses the WikiBrain Java API to query and analyze stored

⁵<http://lucene.apache.org>

⁶Data from <http://stats.grok.se/>

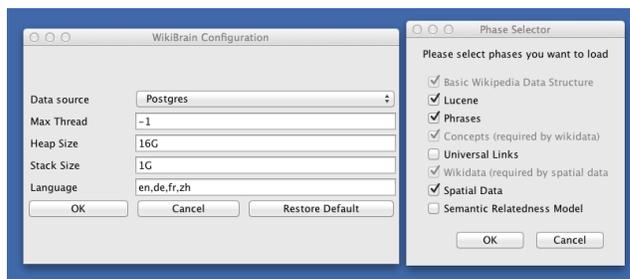


Figure 1: The WikiBrain gui allows supports typical use cases. Advanced users can customize WikiBrain behavior using configuration files.

data. A detailed walkthrough of the API’s use can be seen in Section 7.

The software design of WikiBrain was established with three goals in mind: **flexibility**, **ease of use** and **speed**. These goals reflect our needs and experience developing three applications that mine Wikipedia for knowledge: 1) Omnipedia, a visualization of cultural differences across Wikipedia language editions [2], 2) Atlasify, a geo-spatial visualization framework for Wikipedia concepts [15] and 3) the Macademia research interest visualization website [44]. While we designed WikiBrain with these systems in mind, we believe the three guidelines detailed below serve a wide range of intelligent algorithms and applications.

3.2 Designed for ease of use

WikiBrain strives to reduce the software barriers to Wikipedia data, algorithms, and technologies. To illustrate the ease of integration consider how Jack, a canonical software developer, interacts with the system. Jack needs to prototype an intelligent Wikipedia-based application. He downloads the WikiBrain jar files, runs one Java GUI program (Figure 1) to import Simple English Wikipedia, and adds a few lines of WikiBrain code to his application. For Jack, WikiBrain defaults to using an embedded H2 database⁷; no system dependencies are required. Jack is up and running with Wikibrain and imported Wikipedia data in 20 minutes.

As Jack illustrated, a software developer or researcher can quickly begin their integration of Wikipedia knowledge with little time or effort. Developers can gradually move to more complex modes of interaction with WikiBrain by changing configuration files, loading larger language sets, and using advanced API features.

3.3 Designed for flexibility

We designed WikiBrain to support a wide range of users and applications. To illustrate this flexibility, contrast Jack’s use of WikiBrain with that of Jill, a geographer hoping to use knowledge from Wikipedia in an algorithm to design more semantically cohesive voting districts in the United States. Jill has customized the configuration of WikiBrain using so that it stores data in a PostGIS database server. She relies on state-of-the-art semantic relatedness and Wikipedia spatial layers as foundational algorithms in her work. Using WikiBrain’s evaluation framework she verifies that the SR algorithms she relies upon are performing at expected accuracy levels. She plans to release a reference implemen-

⁷<http://www.h2database.com/html/main.html>

language	articles	links	runtime
Simple English	102K	6M	8 min
German	1.9M	96M	210 min
Full English	4.6M	470M	640 min
25 largest languages	25M	1,670M	3163 min

Table 1: Import times for selected Wikipedia language editions on a 2012 Macbook Pro.

tation of her algorithms that depend on WikiBrain so other researchers can replicate her results.

WikiBrain effectively supports both Jack and Jill. Its architecture allows runtime configuration of components using Java code or JSON configuration files. For example, a WikiBrain-powered web application might use optimized flat file storage of links for SR algorithms, but query the Wikimedia foundation’s live Wikipedia API when showing links to the user.

3.4 Designed for speed

WikiBrain has been engineered for speed on a single machine in both the import and runtime stages. The data import stage uses multi-threaded parsing and loading that enable data from many languages to be loaded quickly.

Table 1 shows import times for selected languages on a 2012 Macbook Pro, excluding the time required to download the database files (which depends on a user’s internet connection speed). The table shows import times for the *core* WikiBrain module that supports pages, redirects, categories, links, multilingual concepts, a Lucene-based full-text search engine, and several other features. Additional WikiBrain modules (e.g. Wikidata, SR, and spatial) can be loaded during the import stage, and require a relatively small amount of time (i.e. less than an hour for full English).

Simple English, a version of Wikipedia written for English learners (and a version that is excellent for the prototyping of intelligent Wikipedia-based applications), loads in 8 minutes on a laptop. The 25 largest Wikipedias, capturing 51 million pages and 1.6 billion links, can be imported in two days.

The runtime API is designed to efficiently support data intensive applications. In general, the speed of query operations scales with the underlying SQL database. However, some critical data is stored in both a representation optimized for speed and in the SQL database. For example, WikiBrain represents wikilinks in both a custom memory-mapped flat file and an SQL table. If a program only needs the articles that link to “Grace Hopper” or vice-versa, the sparse matrix representation is used (it retrieves about a million links per second). If a program wants more detailed information about a wikilink such as its anchortext or location in the article, the SQL table is used.

4. MULTILINGUAL PRIMITIVES

In recent years, much Wikipedia research has analyzed cultural differences in Wikipedia language editions or utilized the encyclopedia as a multilingual resource. Researchers seeking insights into Wikipedia language edition differences have sought to understand the effect of cultural differences in contribution and collaboration patterns [38, 18]) and compared the descriptions of world knowledge in each language edition (e.g. [2, 27]) among other directions (e.g. [53]). Projects that leverage multilingual data from Wikipedia are even more numerous, and have led to innovations in areas

like statistical machine translation (e.g. [46]), topic modeling (e.g. [33]), and bilingual dictionary creation (e.g. [6]).

Despite the diversity of this multilingual Wikipedia research, nearly all work in this space faces the problem of *concept alignment*. In order to treat Wikipedia as a multilingual resource, pages about the same concept in multiple languages must first be “aligned.” For example, an application must group together the article “Germany” in English, “Deutschland” in German, “Alema” in Spanish, and so on. As a result, concept alignment has been a major area of research over the past half-decade.

WikiBrain abstracts this problem from developers. It incorporates a state-of-the-art concept alignment algorithm, *Conceptualign* [2], which can be run during the import stage (assuming the developer has chosen to import more than one language edition).

The world of multilingual Wikipedia has undergone a sea change since Conceptualign was developed in 2011. Beginning with the first stage of the Wikidata project, concept alignment has, in effect, been crowdsourced. These crowdsourced mappings generate the links in the “Languages” sidebar on many Wikipedia pages. As a result of the prominence of Wikidata concept mappings, WikiBrain utilizes these mappings by default in the import stage when more than one language is being processed (rather than those from Conceptualign).

However, Wikidata’s crowdsourcing of concept alignment faces important obstacles — not the least of which involve collaborating across groups of people that speak dozens of different languages — and its concept mappings are not perfect. Many of the most challenging concept mapping cases discussed by Bao et al. [2] are not handled well by Wikidata, e.g. the relationship between “High school” (English Wikipedia), “Secondary school” (English Wikipedia) and their analogues in other languages is problematic. WikiBrain’s concept mapping API has, like its other APIs, been written in a highly extensible format and it is our hope that researchers will utilize WikiBrain as a test bed for improving upon Wikidata’s mappings.

4.1 Using concept mappings

Regardless of the concept mapping strategy used, after building the concept mapping during import, WikiBrain makes it easy to access information at the language-neutral concept level rather than the language-specific article level. For instance, to access all the articles about the concept known in English as “Germany” and print out their page ids, one would only need to write:⁸

```
// UniversalPage represents a multilingual concept.
// It is distinguished from LocalPage, which represents
// language-specific articles.
UniversalPageDao upDao = configurator.get(
    UniversalPageDao.class);
UniversalPage germany = upDao.getByName(
    "Germany", Language.EN);
for (LocalId id : germany.getLanguageSpecificEntities()){
    System.out.println(id);
}
```

The ability to use language-neutral Wikipedia concepts has large implications. Research has shown that each language edition of Wikipedia represents vastly different views

⁸This and all other source code listings reflect the WikiBrain API at the time of writing. For up to date information, please visit <http://wikibrainapi.org>

of encyclopedic world knowledge. Even many of the small language editions have a great deal of content not in any other language edition. As such, concept-level access to Wikipedia unlocks significantly more world knowledge for system builders than any single language edition. In addition, for those who study multilingual Wikipedia, concept-level access to Wikipedia information will allow straightforward comparisons of any Wikipedia resource supported by WikiBrain (e.g. comparing the number of page views received by articles about the same concept in different languages, etc.).

5. SEMANTIC-RELATEDNESS PRIMITIVES

This section details how WikiBrain moves beyond explicit semantic relationships to algorithmically infer the strength of relationships between words, phrases, and articles using semantic relatedness (SR) algorithms.

The automatic estimation of the relatedness between two concepts has been an active area of research in artificial intelligence (AI) and natural language processing (NLP) for decades [43]. Semantic relatedness (SR) algorithms - the family of algorithms that perform this estimation - underly an enormous variety of applications in a series of domains. These applications range from low-level tasks like word sense disambiguation [36] and coreference resolution [40] to high-level technologies like search engines [42] and information visualization systems [3].

Generally speaking, SR algorithms provide a single numeric estimate of the number and strength of relationships between any two concepts a and b (typically between 0 and 1) [18]. For instance, if concept a is “human-computer interaction” and concept b is “Alan Kay”, we might expect a good semantic relatedness algorithm to output a high value for $SR(a, b)$. Conversely, if concept b is replaced with “Kim Kardashian”, we might expect the opposite.

Researchers have proposed literally hundreds of SR algorithms over the past two decades. All state-of-the-art SR algorithms require a large human-engineered knowledge base. For some algorithms, this knowledge base must be Wikipedia for its link, category, or concept structure [32, 5]. Other SR algorithms require a text corpora such as Wikipedia [11]. Thus, most modern SR algorithms can be built using Wikipedia as a knowledge base.

Though SR algorithms have existed for decades, they present serious implementation challenges to researchers and practitioners. SR algorithms typically require large corpora of data that must be carefully analyzed. It is also difficult to find robust, efficient implementations of SR algorithms. These challenges hurt not only industry developers, but researchers developing visualization, NLP, or IR algorithms that use SR as an algorithmic primitive.

WikiBrain fills this gap by providing a robust end-to-end SR framework that allows developers and research to easily tap state-of-the-art SR algorithms. WikiBrain’s SR API extends beyond the traditional single `SR("apple", "orange")` method to offer three SR methods critical to real-world applications (Figure 2). First, traditional pairwise SR estimates can be obtained for both phrases and Wikipedia articles. Second, given a target phrase or article, WikiBrain efficiently retrieves the highest scoring articles (`mostSimilar`). Third, WikiBrain calculates the cosimilarity matrix between articles or phrases. Applications and researchers can combine these SR primitives to build applications that reflect

```
// Estimate similarity between phrases or articles
SRResult similarity(String phrase1, String phrase2);
SRResult similarity(int pageId1, int pageId2);

// Find the most similar articles to a phrase or article.
SRResultList mostSimilar(int pageId);
SRResultList mostSimilar(String phrase);

//The cosimilarity matrix between phrases or articles.
float[][] cosimilarity(String phrases[]);
float[][] cosimilarity(int pageIds[]);
```

Figure 2: A listing of the SR primitives provided by WikiBrain. Only the first function is typically supported by SR implementations, but applications often need a richer API.

human intuition will still running efficiently.

WikiBrain SR has been optimized to support the performance needs of real-world applications. SR algorithms are traditionally designed to efficiently estimate the relatedness of a pair of words. However, intelligent systems often need to calculate hundreds or thousands of estimates to perform a task. For example, a search engine wanting to display Wikipedia articles related to a query would estimate the relatedness of a query to each of the millions of Wikipedia articles and return the top scoring articles. WikiBrain has been optimized for demanding scenarios such as these. As an example, WikiBrain can calculate the pairwise semantic estimates between all 4 million English Wikipedia pages (roughly 8 trillion SR estimates) in a few days.⁹

WikiBrain includes a rich set of six well-tested SR algorithms that developers interact with through a simple unified interface (Figure 2). These SR algorithms analyze Wikipedia’s category structure [47], link structure [32, 15], and article text [5, 30] to generate SR estimates. In addition to these five SR implementations, WikiBrain supports an SR “ensemble” that fits a linear combination of other SR algorithms as described in [15].

WikiBrain also provides a robust SR evaluation framework. It includes reference gold-standard datasets, cross-validation support, and common subcomponents needed for SR such as vector models and disambiguators. WikiBrain includes evaluation support for the traditional SR accuracy measures between SR estimates and gold-standards (mean-absolute-error, Pearson’s correlation, and Spearman’s correlation). It also provides a framework for evaluating information-retrieval metrics such as precision and recall — characteristics important to many real world systems, but generally not studied by the SR community.

5.1 Example SR use

We now demonstrate WikiBrain SR through a short case study. We first ask WikiBrain for the English articles most closely related to the phrase “jazz:”

```
SRResultList similar = sr.mostSimilar("jazz", 5);
```

Printing out the resulting list, we would see:

```
0.908: Jazz
0.784: Blues
0.760: American popular music
0.757: American folk music
0.754: Rock and roll
```

⁹This computation calls `mostSimilar` for each Wikipedia article with a large result limit (e.g. 100,000 articles), effectively truncating the near-zero SR estimates at 0.0.

We then restrict results to just movies (Wikipedia concept id 11424). We first retrieve all the Wikipedia articles that are movies.

```
Set<LocalId> candidates = wikiDataDao.pagesWithValue(
    "instance of", WikidataValue.forItem(11424),
    Language.EN);
```

We would then convert the local ids to a set of integers called candidates and pass these to the mostSimilar method.

```
SRResultList results = sr.mostSimilar("jazz", 10,
    candidates);
```

Resulting in a list of movies related to "jazz:"

```
0.785: Soul to Soul
0.776: All You Need Is Love: The Story of Popular Music
0.766: One Night with Blue Note
0.761: The World According to John Coltrane
0.760: Paris Blues
```

6. SPATIAL PRIMITIVES

Recognizing that Wikipedia contains an unprecedented amount of structured information about places in the real world, researchers and practitioners in geographic information science, geography and computer science have leveraged Wikipedia in numerous research projects and products. Indeed, Wikipedia articles about geographic entities — often called “geographic articles” [17] — have become a canonical example of *volunteered geographic information*, the subset of user-generated content that contains a geographic reference and that has been described as having “profound impacts on...the discipline of geography and its relationship to the general public.” [9].

Researchers have utilized geographic Wikipedia articles in a wide range of studies. For instance, Hecht and Gergle [16] leveraged the geographic information in Wikipedia to examine the variation in spatial coverage of user-generated content, and Graham and colleagues have built on this work to describe the “geographies of world knowledge” as described in Wikipedia (as well as other web content) [10]. Other geographic research utilizing Wikipedia data has included several examinations of the relationship between a user’s location and the locations about which they submit user-generated content (e.g. [17, 26, 13]), geographic studies of controversy in Wikipedia [52], and building geographic text models with Wikipedia data [51].

Outside the research world, geographic information in Wikipedia is even more prominent. Text from the corresponding geographic article is displayed prominently alongside a map when searching for a place name on Google and Bing. In addition, Wikipedia has become an important tool for connecting place names to their corresponding geographic coordinates across the Web (e.g. BBC [24]).

However, each of these projects and products uses its own custom infrastructure for processing and utilizing Wikipedia as a source of geographic information. As with semantic relatedness and multilingual concept alignment, reproducibility, comparability, and the ability to leverage advances in new applications has been limited. Any researcher or developer wishing to take advantage of the geographic information in Wikipedia must start at a low level, reinventing the wheel and resolving (or ignoring) important problems.

WikiBrain addresses this situation by abstracting the process of gathering, processing, and cleaning geographic infor-

mation in Wikipedia. Any user of WikiBrain can immediately begin to leverage this information in novel applications with no start-up costs other than a brief, automated import stage. Specifically, WikiBrain supports the following four capabilities, all of which play important roles in geographic Wikipedia research and applications:

1. Easy access to Wikipedia article “geotags.”
2. Solutions to the Geoweb Scale Problem, a major obstacle in the use of Wikipedia’s geographic information.
3. Connections to well-known libraries for processing geographic information, thus reducing common statistical errors and computational inefficiencies found in geographic Wikipedia research.
4. Generalization to non-geographic spatial domains, an emerging area of research and technologies.

The design of these capabilities is covered in detail in the following sections.

6.1 Latitude and longitude coordinates

Geographic analyses and applications based on Wikipedia are, at a core level, enabled by the large number of articles that have been tagged with latitude and longitude coordinates (i.e. “geotagged”). For instance, the article “Berlin” in the English Wikipedia is tagged with the coordinate (13.38,52.51) and the article “Alaska” in the English Wikipedia is tagged with (-150,64). These tags occur either through templates, or, more recently, as part of Wikidata. While historically researchers and practitioners had to parse these templates themselves, pre-parsed datasets of article-linked coordinates have been available for a number of years (e.g. DBpedia’s Geo-Coordinates dataset¹⁰ and Wikipedia-World¹¹). Wikidata provides a means for users to independently obtain up-to-date datasets of these coordinates.

After automated import, developers using WikiBrain can easily access latitude and longitude coordinates that are linked to Wikipedia articles using a single line of code:

```
// get latitude and longitude coordinate by article name
Geometry g = spatialDao.getGeometry("Alaska", Language.EN
    , "wikidata");

// get latitude and longitude point by Wikidata item ID,
// using static shortcuts for built-in layers
Geometry g = spatialDao.getGeometry(797, Layers.WIKIDATA);
```

The “wikidata” in the above code refers to the “wikidata” layer, which contains all of the geographic coordinates in Wikidata. WikiBrain also includes layers that contain information from DBpedia and Natural Earth.¹²

Even though the days of parsing geographic coordinates oneself are fortunately over, several large challenges still make it difficult to utilize Wikipedia as a source of geographic information. Moreover, these challenges are not nearly as visible as the need to extract coordinates through parsing and, as a result, they can — and have — caused important silent failures. Through abstraction and connection with standard open-source geographic tools and datasets, WikiBrain makes it straightforward to address the most important of these problems: (1) the Geoweb Scale Problem

¹⁰<http://wiki.dbpedia.org/Datasets>

¹¹http://de.wikipedia.org/wiki/Wikipedia:WikiProjekt_Georeferenzierung/Hauptseite/Wikipedia-World/en

¹²<http://naturalearthdata.com>

and (2) the proper basic handling of geographic information. These are the subjects of the next two sections.

6.2 The Geoweb Scale Problem

The Geoweb Scale Problem occurs when spatial representations are “at too coarse a scale for a given problem”[19]. As noted above, all spatial representations in Wikipedia are single coordinates, the coarsest possible representation of a spatial entity. For instance, Berlin, which has a spatial footprint of almost 900 square kilometers, is represented as a single point at the center of the city.

Any geographic research project or product that requires consideration of distance, spatial containment, neighbors (next-to relations), and many other geographic operations will incur significant inaccuracies when using point representations. For instance, consider Hecht and Gergle’s work that sought to evaluate the “localness” of the editors of geographic Wikipedia articles [17]. Had they not corrected for the Geoweb Scale Problem, they would have incorrectly assessed the distance between an editor in Alaska’s capital, Juneau, and the state of Alaska, to be over 1000km, and thus considered the editor to be very “non-local”.

WikiBrain has easy-to-use implementations of all three methods researchers have used to address the Geoweb Scale Problem in Wikipedia in previous literature. Namely, WikiBrain (1) includes rich spatial representations for all countries and first-order administrative districts (e.g. states, provinces, Bundesländer), (2) it allows for automated connection of Wikipedia articles to rich spatial representations in a Shapefile format (an industry standard), and (3) it affords the easy filtering out of very large spatial entities.

WikiBrain automatically connects global country and first-order administrative district shapefiles from the NaturalEarth project to their corresponding Wikipedia articles during the spatial import stage using title-matching and pre-executed manual correction. These shapefiles contain detailed *polygonal* spatial representations of countries and administrative districts, thereby eliminating the Geoweb Scale Problem in these important cases. To access one of these rich spatial representations, one must simply write one of the following single lines of code:

```
// get geometries by article name and language;
// GADM1 refers to first-order districts,
// GADM0 refers to countries
Geometry g = spatialDao.getGeometry("Alaska", Language.
    getByLangCode("en"), "state");

// get geometry by Wikidata item ID
Geometry g = spatialDao.getGeometry(797, Layers.STATE);
```

Wikipedia-based products and research projects may require rich geographic representations for other types of geographic Wikipedia articles (e.g. polygonal representations of cities, counties, national parks, etc.). As such, WikiBrain makes it easy to connect representations in arbitrary shapefiles to Wikipedia articles. To do so, WikiBrain utilizes named entity disambiguation and a powerful geo-spatial database matching ruleset.

Finally, certain research projects address the the Geoweb Scale Problem by eliminating geographic Wikipedia articles with a sufficiently large spatial footprint (e.g. [26]). To do so, they have leveraged a convention in Wikipedia point representations in which larger entities specify coordinates in less precision. For instance, Alaska’s coordinate specifies degrees latitude and longitude, but not minutes or seconds.

To support this approach, WikiBrain allows users to specify a minimum precision level when obtaining a geometry. For instance, after executing the following code, *g* would be null.

```
Geometry g = spatialDao.getGeometry("Alaska", Language.
    getByLangCode("en"), "wikidata", LatLonPrecision.
    HIGH)
```

WikiBrain also supports a variant of this approach that has appeared in the literature (e.g. [20]) in which just countries and first-order administrative districts are removed (see below for an example).

6.3 Geographic information best practices

Research and systems that use Wikipedia as a source of information about the real world are often developed by computer scientists with little background in the storing, managing, processing, and communicating of geographic information [50, 21]. As such, WikiBrain provides simple access to geographic information best practices for WikiBrain developers. To do so, we built WikiBrain’s spatial capabilities on top of the GeoTools¹³ software library, the most well-known open-source Java geographic information systems software library. Indeed, the *Geometry* data type you see above belongs to this library.

Through our GeoTools-based approach, WikiBrain users have easy access to powerful geographic operations. For instance, rather than calculating distances in degrees latitude and longitude – a common but highly incorrect practice in the processing of Wikipedia geographic information – WikiBrain users can leverage GeoTools’ GeodeticCalculator, which can perform distance calculations on highly sophisticated models of the earth in just a few lines of code (see case study in Section 7 for an example).

6.4 Non-geographic spatial reference systems

In recent years the use of geographic operations in non-spatial references has gained prominence. For instance, researchers have utilized geographic information systems (GIS)-like techniques on periodic tables [15], basketball courts [8], and data-driven spaces that are the result of dimension reduction [45]. Wikipedia has played a key role in several research projects in this area (e.g. [15, 35]) as many of these projects require a robust text model for each entity in a non-geographic reference system.

WikiBrain makes this type of research much easier. WikiBrain’s spatial capabilities have been designed from the ground up to support non-geographic reference systems. For example, the following code retrieves the geometry for “Sodium” in WikiBrain’s built-in periodic table spatial reference system (one of several built-in non-geographic reference systems from [15]):

```
Geometry g = spatialDao.getGeometry("Sodium", Language.EN
    , "elements", "periodic_table");
```

7. CASE STUDY: TOBLER’S FIRST LAW

The previous sections describe WikiBrain’s capabilities at a high level. In order to provide a lower-level understanding of the software library, this section walks through a case study of its use. In keeping with our view of WikiBrain as a means of democratizing access to state-of-the-art Wikipedia-based research, the case study involves the replication of a

¹³<http://www.geotools.org>

Wikipedia research project by Hecht and Moxley [20] that provided the first empirical examination of what is known as *First Law of Geography*, or *Tobler's First Law* (TFL) [48]. The case study shows how the thousands of lines of code that were required for the initial study are reduced to only a few lines using WikiBrain, and how many of the capabilities described above (e.g. spatial, semantic relatedness, and multilingual) can be easily employed and integrated.

Tobler's First Law is a well-known precept in the field of geography that states that "everything is related to everything else, but near things are more related than distant things." [48]. While TFL provides the theoretical basis for many large research areas within geography and related domains, prior to the work of Hecht and Moxley, it had not been validated quantitatively across general world knowledge. Wikipedia-based semantic relatedness measures applied to geographic Wikipedia articles examined across multiple languages allowed Hecht and Moxley to perform this evaluation.

We now describe how a developer would replicate Hecht and Moxley's work using WikiBrain.

7.1 Case study walk-through

To replicate Hecht and Moxley, a developer would begin by integrating WikiBrain into a new project by either downloading pre-built JARs from the WikiBrain site or referencing WikiBrain as a maven dependency as shown below.

```
<dependency>
  <groupId>org.wikibrainapi</groupId>
  <artifactId>wikibrain</artifactId>
  <version>0.3.0</version>
</dependency>
```

Since this analysis uses the large full English Wikipedia, the developer would configure WikiBrain to use the scalable Postgres database instead of the (default) embedded h2, which is useful for smaller language editions (e.g. Simple English). They would create an "override" configuration file, which we name "my.conf".

```
dao.dataSource.default : psql
dao.dataSource.psql {
  username : tobler
  password : ""
  url : "jdbc:postgresql://localhost/wikibrain"
}
```

The developer would next import Wikipedia data by running Java classes supplied by WikiBrain. WikiBrain classes using the WikiBrain import GUI (Figure 1), a Java IDE such as Eclipse and IntelliJ, or by running WikiBrain's `wb-java.sh` utility shell script that appropriately sets the Java classpath.

```
# load basic data for English and German using our config
org.wikibrain.dao.Loader -l en,de -c my.conf \
  -s core -s spatial

# Train the SR algorithm in both German and English
org.wikibrain.sr.SRBuilder -c my.conf -l en
org.wikibrain.sr.SRBuilder -c my.conf -l de
```

The first command imports core (categories, pages, links, etc.) and spatial Wikipedia data in English and German using our configuration file. The final commands train the ensemble SR algorithm in English and German. Although Hecht and Moxley performed their study on 25 large language editions, sufficient semantic relatedness ground truth data does not exist outside English and German, so we focus on these language editions in this case study.

Once the commands have finished, the developer is ready to begin writing her Java program. Below, we highlight the key parts of this program, but the full code can be found on the WikiBrain website.¹⁴

All WikiBrain programs begin by creating an "Environment" that provides access to WikiBrain components. The environment is created using the command line args so that the program can accept standard WikiBrain options such as a configuration file ("-c my.conf") and a language ("-l en").

Replicating Hecht and Moxley's study requires access to several key WikiBrain components. Programs retrieve components by asking the WikiBrain environment's "configurator" for a component of that class. WikiBrain uses the Data Access Object (DAO) pattern¹⁵ for components that import and query data. Our program needs DAOs for language-specific pages (called LocalPages), multilingual concepts (called UniversalPages), and spatial data. It will also need SR algorithms (called MonolingualSRMetric) for both English and German.

```
// Create WikiBrain environment from command-line args
// This could be customized in Java code as well.
Env env = EnvBuilder.envFromArgs(args);

// Get data access objects from the configurator
Configurator c = env.getConfigurator();
LocalPageDao lpd = c.get(LocalPageDao.class);
UniversalPageDao upd = c.get(UniversalPageDao.class);
SpatialDao sd = c.get(SpatialDataDao.class);

// Get SR algorithms in German and English
SRMetric deSr = c.get(SRMetric.class, "ensemble",
  "language", "de");
SRMetric enSr = c.get(SRMetric.class, "ensemble",
  "language", "en");
```

We next collect the spatial representations for all geographic concepts that have articles in both English and German. For instance, the following code will return a concept id and geometry for Berlin, San Francisco, the Bundestag, etc. Note that, as described above, we filter out first-order administrative districts and countries, which was also done by Hecht and Moxley.

```
LanguageSet langs = new LanguageSet("en,de");
Map<UniversalPage, Point> locations = new HashMap<>();

// Get all geometries and their corresponding concept
// IDs for geographic articles not in the country or
// state layers
Map<Integer, Geometry> geo = sd.getAllGeometries(
  Layers.WIKIDATA, Layers.COUNTRY, Layers.STATE);

// Add points for all concepts that have articles
// in both the German and English Wikipedias
for (Integer conceptId : geo.keySet()) {
  UniversalPage concept = upd.getById(conceptId);
  if (concept.hasAllLanguages(langs)) {
    locations.put(concept,
      (Point)geo.get(conceptId));
  }
}
```

Finally, we randomly sample pairs of concepts (code not shown). Given two concepts, `c1` and `c2`, we calculate the geodetic distance and estimate the relatedness in both German and English.

¹⁴<https://github.com/shilad/wikibrain/blob/master/wikibrain-spatial/src/main/java/org/wikibrain/spatial/cookbook/SimpleToblersLawEvaluator.java>

¹⁵<http://www.oracle.com/technetwork/java/dataaccessobject-138824.html>

```

// Get points from earlier.
Point p1 = (Point)locations.get(c1);
Point p2 = (Point)locations.get(c2);

// Compute distance in kms between points
GeodeticCalculator calc = new GeodeticCalculator();
calc.setStartingGeographicPoint(p1.getX(), p1.getY());
calc.setDestinationGeographicPoint(p2.getX(), p2.getY());
double km = calc.getOrthodromicDistance() / 1000;

// Compute SR estimates in English and German
SRResult sr1 = enSr.similarity(
    c1.getLocalId(Language.EN),
    c2.getLocalId(Language.EN));
SRResult sr2 = deSr.similarity(
    c1.getLocalId(Language.DE),
    c2.getLocalId(Language.DE));

```

Figure 3 shows the results for 100,000 sampled concept pairs. As Tobler’s First Law predicts (and Hecht and Moxley found), when distance increases, relatedness decreases. However, while Hecht and Moxley had to develop their own approach to accessing geographic information in Wikipedia, their own multilingual alignment approaches, and their own semantic relatedness measure, the above case study only required a few lines of code. In addition, the above example uses state-of-the-art semantic relatedness measures, which was not true of the Hecht and Moxley study.

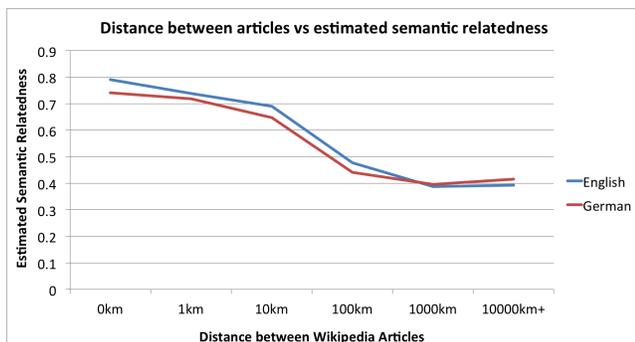


Figure 3: The relationship between the distances between all geographic articles and their estimated semantic relatedness. Both English and German show similar patterns.

8. CONCLUSION AND FUTURE WORK

In this paper, we have described WikiBrain, a new extensible software framework that democratizes access to a series of intelligent technologies that derive their world knowledge from Wikipedia. WikiBrain provides the first evaluation testbed for some of these technologies with the goal of accelerating and improving research through improved comparability. It is our hope that developers of new Wikipedia-based intelligent technologies (and existing technologies not yet implemented in WikiBrain) will choose WikiBrain to do so. In that vein, WikiBrain also provides fast, easy-to-use, and flexible access to the majority of basic structures that have proven useful in research on Wikipedia-based intelligent technologies (e.g. links, categories, page views, Wiki-data).

While we have designed WikiBrain for those who wish to use Wikipedia rather than study it, researchers interested in the content, communication, and collaboration patterns of Wikipedia may also find WikiBrain useful. For instance, WikiBrain is well situated to assist researchers interested

in comparing different language editions of Wikipedia. In addition, the intelligent Wikipedia-based technologies built into WikiBrain could help reveal new insight about Wikipedia, e.g. better understanding the category network with semantic relatedness measures. Moving forward, we have a team in place for summer 2014 to add robust support for edit histories (and related features like quality ranking) and information about users). In addition to democratizing access to additional resources that have proven useful in intelligent technologies, these features may make WikiBrain much more appealing to those who study Wikipedia.

Finally, it is important to note that while this paper described many of the features of WikiBrain, many additional features exist in the software. WikiBrain was developed by several long-time Wikipedia researchers, and we designed it to make it easier to address many of the very low-level challenges faced by those who work with Wikipedia data. For instance, WikiBrain allows combined or separate access to “parseable” links (i.e. those that appear directly in the XML dumps) and “unparseable” links (i.e. those that are hidden in templates), an issue not even considered by most developers of Wikipedia-based intelligent technologies, even those that rely heavily on the link graph. We encourage readers to explore WikiBrain themselves by downloading the software framework and using it in their applications and research. Perhaps more importantly, we also encourage readers to help improve WikiBrain by joining our team of developers on GitHub.

9. ACKNOWLEDGEMENTS

This research has been generously supported by Macalester College, 3M’s Non-Tenured Faculty Award Program, and the National Science Foundation (grant IIS-0964697). We would like to thank Patti Bao, Darren Gergle, Samuel Carton, and Mahmood Quaderi for their contributions to predecessors of WikiBrain. We also acknowledge other software projects that have supported and inspired WikiBrain; particularly GitHub, JWLP, OpenGeo, NaturalEarth, and TravisCI. Most of all, we want to thank Wikipedians. Our work, and literally hundreds of NLP, AI, and IR algorithms would not have been possible without the billions of words of knowledge you encoded within Wikipedia.

10. REFERENCES

- [1] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. DBpedia: A Nucleus for a Web of Open Data. *Lecture Notes in Computer Science*, page 722–735, 2007.
- [2] P. Bao, B. Hecht, S. Carton, M. Quaderi, M. Horn, and D. Gergle. Omnipedia: bridging the wikipedia language gap. In *CHI '12*, 2012.
- [3] T. Bergstrom and K. Karahalios. Conversation clusters: grouping conversation topics through human-computer dialog. In *CHI '09*, pages 2349–2352, Boston, MA, 2009.
- [4] S. Cucerzan. Large-scale named entity disambiguation based on wikipedia data. In *EMNLP-CoNLL*, volume 7, pages 708–716, 2007.
- [5] O. Egozi, S. Markovitch, and E. Gabrilovich. Concept-Based Information Retrieval Using Explicit Semantic Analysis. *Trans. Inf. Syst.*, 29(2):1–34, 2011.
- [6] M. Erdmann, K. Nakayama, T. Hara, and S. Nishio. An approach for extracting bilingual terminology from wikipedia. In *Database Systems for Advanced Applications*, pages 380–392. Springer Berlin Heidelberg, Jan. 2008.

- [7] E. Gabrilovich and S. Markovitch. Wikipedia-based semantic interpretation for natural language processing. *JAIR*, 34:443–498, 2009.
- [8] K. Goldsberry. CourtVision | examining the NBA through spatial and visual analytics, 2012.
- [9] M. F. Goodchild. Citizens as sensors: the world of volunteered geography. *GeoJournal*, 69(4):211–221, 2007.
- [10] M. Graham, S. A. Hale, and M. Stephens. *Geographies of the World's Knowledge*. Convoco! Edition, 2011.
- [11] G. Halawi, G. Dror, E. Gabrilovich, and Y. Koren. Large-scale learning of word relatedness with constraints. In *KDD '12*, 2012.
- [12] A. Halfaker. MediaWiki utilities.
- [13] D. Hardy, J. Frew, and M. F. Goodchild. Volunteered geographic information production as a spatial process. *IJGIS*, 26(7):1191–1212, 2012.
- [14] S. Harispe, S. Ranwez, S. Janaqi, and J. Montmain. Semantic measures for the comparison of units of language, concepts or entities from text and knowledge base analysis. *CoRR*, abs/1310.1285, 2013.
- [15] B. Hecht, S. H. Carton, M. Quaderi, J. Schöning, M. Raubal, D. Gergle, and D. Downey. Explanatory semantic relatedness and explicit spatialization for exploratory search. *SIGIR '12*, 2012.
- [16] B. Hecht and D. Gergle. Measuring self-focus bias in community-maintained knowledge repositories. In *C&T '09*, page 11–19, 2009.
- [17] B. Hecht and D. Gergle. On the "Localness" of user-generated content. In *CSCW '10*, 2010.
- [18] B. Hecht and D. Gergle. The tower of babel meets web 2.0: User-generated content and its applications in a multilingual context. In *CHI '10*. ACM, 2010.
- [19] B. Hecht and D. Gergle. A beginner's guide to geographic virtual communities research. IGI Global, 2011.
- [20] B. Hecht and E. Moxley. Terabytes of toblor: evaluating the first law in a massive, domain-neutral representation of world knowledge. In *COSIT '09*, 2009.
- [21] B. Hecht, J. Schöning, L. Capra, A. Mashhadi, L. Terveen, and M.-P. Kwan. 2013 workshop on geographic human-computer interaction. In *CHI '13 EA.*, 2013.
- [22] J. Hoffart, F. M. Suchanek, K. Berberich, and G. Weikum. Yago2: A spatially and temporally enhanced knowledge base from wikipedia. *Artificial Intelligence*, 194:28–61, 2013.
- [23] A. Kittur, E. H. Chi, B. A. Pendleton, B. Suh, and T. Mytkowicz. Power of the few vs. wisdom of the crowd: Wikipedia and the rise of the bourgeoisie. In *CHI '07*, 2007.
- [24] G. Kobilarov, T. Scott, Y. Raimond, S. Oliver, C. Sizemore, M. Smethurst, C. Bizer, and R. Lee. Media meets semantic web – how the BBC uses DBpedia and linked data to make connections. In *The Semantic Web: Research and Applications*, number 5554 in Lecture Notes in Computer Science, pages 723–737. Springer Berlin Heidelberg, 2009.
- [25] S. Lam, A. Uduwage, Z. Dong, S. Sen, D. Musicant, L. Terveen, and J. Riedl. WP:Clubhouse? an exploration of wikipedia's gender imbalance. In *WikiSym '11.*, 2011.
- [26] M. D. Lieberman and J. Lin. You are where you edit: Locating wikipedia users through edit histories. In *ICWSM '09*, 2009.
- [27] P. Massa and F. Scrinzi. Manypedia: Comparing language points of view of wikipedia communities. In *WikiSym '12*, 2012.
- [28] D. J. McIver and J. S. Brownstein. Wikipedia usage estimates prevalence of influenza-like illness in the united states in near real-time. *PLoS Comput Biol*, 10(4):e1003581, Apr. 2014.
- [29] M. Mestyán, T. Yasseri, and J. Kertész. Early prediction of movie box office success based on wikipedia activity big data. *PLoS ONE*, 8(8):e71226, Aug. 2013.
- [30] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.
- [31] R. Miller. Wikipedia founder jimmy wales responds. *Slashdot: News for Nerds, Stuff That Matters*, 28, 2004.
- [32] D. Milne and I. Witten. An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. In *Proceeding of AAAI Workshop on Wikipedia and Artificial Intelligence: an Evolving Synergy*, 2008.
- [33] D. Minmo, H. M. Wallach, J. Naradowsky, D. A. Smith, and A. McCallum. Polylingual topic models. In *EMNLP '09*, 2009.
- [34] C. Okoli, M. Mehdi, M. Mesgari, F. Nielsen, and A. Lanamäki. The people's encyclopedia under the gaze of the sages: A systematic review of scholarly research on wikipedia. *Available at SSRN*, 2012.
- [35] C. Pang and R. Biuk-Aghai. Wikipedia world map: Method and application of map-like wiki visualization. In *WikiSym '11*, Mountain View, CA, 2011.
- [36] S. Patwardhan, S. Banerjee, and T. Pedersen. Using measures of semantic relatedness for word sense disambiguation. In *CICLING '03*, 2003.
- [37] T. Pedersen, S. Patwardhan, and J. Michelizzi. Wordnet:: Similarity: measuring the relatedness of concepts. In *Demonstration Papers at HLT-NAACL 2004*, 2004.
- [38] U. Pfeil, P. Zaphiris, and C. S. Ang. Cultural differences in collaborative authoring of wikipedia. *JCMC*, 12(1):88–113, Oct. 2006.
- [39] G. Pirró. Reword: Semantic relatedness in the web of data. In *AAAI '12*, 2012.
- [40] S. P. Ponzetto and M. Strube. Exploiting semantic role labeling, WordNet and wikipedia for coreference resolution. In *NAACL '06*, 2006.
- [41] R. Priedhorsky, J. Chen, S. T. K. Lam, K. Panciera, L. Terveen, and J. Riedl. Creating, destroying, and restoring value in wikipedia. In *Group '07*, 2007.
- [42] K. Radinsky, E. Agichtein, E. Gabrilovich, and S. Markovitch. A word at a time: computing word relatedness using temporal semantic analysis. In *WWW '11*, pages 337–346. ACM, 2011.
- [43] P. Resnick. Using information content to evaluate semantic similarity in a taxonomy. In *IJCAI '95*, 1995.
- [44] S. Sen, E. Nunes, E. I. Sparling, H. Charlton, R. Kerwin, J. Lim, B. Maus, N. Miller, M. R. Naminski, A. Schneeman, and et al. Macademia. *IUI '11*, 2011.
- [45] A. Skupin and S. I. Fabrikant. Spatialization methods: A cartographic research agenda for non-geographic information visualization. *CAGIS*, 30(2):95–115, 2003.
- [46] J. R. Smith, C. Quirk, and K. Toutanova. Extracting parallel sentences from comparable corpora using document level alignment. In *NAACL '10*, 2010.
- [47] M. Strube and S. P. Ponzetto. WikiRelate! computing semantic relatedness using wikipedia. In *AAAI '06*, 2006.
- [48] W. R. Tobler. A computer movie simulating urban growth in the Detroit region. *Economic geography*, 1970.
- [49] D. Vrandečić. Wikidata: A New Platform for Collaborative Data Collection. In *WWW '12 Companion*, 2012.
- [50] M. Wiesmann. Falsehoods programmers believe about geography, 2012. 00000.
- [51] B. P. Wing and J. Baldrige. Simple supervised document geolocation with geodesic grids. In *ACL '11*, 2011.
- [52] T. Yasseri, A. Spoorri, M. Graham, and J. Kertész. The most controversial topics in wikipedia: A multilingual and geographical analysis. In P. Fichman and N. Hara, editors, *Global Wikipedia: International and cross-cultural issues in online collaboration*. Scarecrow Press, 2014.
- [53] T. Yasseri, R. Sumi, and J. Kertész. Circadian patterns of wikipedia editorial activity: A demographic analysis. *PLoS One*, 7(1):1–8, Jan. 2012.