2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)

Witt: Querying Technology Terms based on Automated Classification

Mathieu Nassif
School of Computer Science
McGill University
Montréal, QC, Canada
mnassif@cs.mcgill.ca

Christoph Treude
School of Computer Science
University of Adelaide
Adelaide, Australia
christoph.treude@adelaide.edu.au

Martin P. Robillard
School of Computer Science
McGill University
Montréal, QC, Canada
martin@cs.mcgill.ca

***Abstract*—Witt is a tool that systematically and automatically categorizes software technologies using original information extraction algorithms applied to Stack Overflow and Wikipedia. Witt takes as input a term, such as "django", and returns one or more categories that describe it (e.g., "framework"), along with attributes that further qualify it (e.g., "web-application"). Our comparative evaluation of Witt against six independent taxonomy tools showed that, when applied to software terms, Witt has better coverage than alternative solutions, without a corresponding degradation in the number of spurious results. The information extracted by Witt is available through the Witt Web Application, which allows users to query and explore Witt's categorization of software technologies by both obtaining the category for a term, and all the terms in a given category.**
On-line Portal: https://cs.mcgill.ca/~swevo/witt-web
Video: https://www.youtube.com/watch?v=tPsp1M4Ua3w

***Index Terms*—Taxonomy tool, Hypernym detection, Web application.**

I. Introduction

Software development increasingly relies on libraries and frameworks that support an ever-expanding constellation of functionalities, from classic data storage services to the latest flavor of code generation for web application interfaces. In fact, one could argue that in the last decade we have witnessed a Cambrian explosion [1] of sorts in the software development domain, with each day bringing the release of new technologies. This wide availability of tools and reusable components is of course a boon for prospective users, if and when they find a reliable solution to their needs. The catch, however, is that it is easy to get lost in this giant bazaar of software componentry [2]. What software developer has never asked the question: *"what is this technology?"*

The decentralized nature of current-day development creates many obstacles for anyone seeking to understand the software technology landscape. Most notably, the absence of an official nomenclature and the reuse of terms from other domains create ambiguity. Often, the amount of contextual information necessary to clear up the ambiguity adds up to a definition of the term of interest. For example, consider a technology named Snap. Searching the Internet for the single term "Snap" is of course fruitless. Adding disambiguating terms such as "software" or "programming" is also not effective given the plethora of software systems called "Snap" and variants. A very dedicated searcher could consider approaches that are more sophisticated than a general-purpose Internet search. For example, one could use taxonomy tools such as WordNet [3] or WebIsADb [4]. However, previous work has already demonstrated the poor performance of such tools when used in the context of software development [5], [6]. It is also possible to search specific resources directly, such as Wikipedia disambiguation pages [7] or the Stack Overflow tag catalog [8], but this requires additional effort and intuition. Ultimately, to discover the correct technology classification for Snap in a given context, it is necessary to already know what we are looking for: for instance, to search for Snap Web Framework.

Categorizing software technologies is an expression of the hypernym discovery problem, which attempts to associate a term T with a hypernym H by the relation "T *is a type of* H". The difficulty of automatically detecting hypernyms for software technologies means that the inverse function, hyponymy, is even more challenging to establish. In our context, hyponymy lists all technologies within a category, e.g., all web application frameworks. This is a potentially even more valuable piece of information for developers and project managers, as it would allow them to systematically review the software technology landscape, for example to decide on which technologies to adopt or support. Unfortunately, listing all technologies of a certain type is currently mostly supported through crowd-sourcing, which is prone to rapid obsolescence and errors of omission. For example, at the time of writing, the Wikipedia page "Comparison of web frameworks", which contributes a list of over 100 web application frameworks, bears the mention "*This article needs to be updated. Please update this article to reflect recent events or newly available information. (December 2015)*" [9].

We developed a tool for the *systematic and automated* categorization of software technologies. Our approach, called Witt, for *What Is This Technology?*, takes as input a term such as Snap, can show available variants, such as snap-framework and returns one or more general categories that describe it (e.g., *framework*), along with attributes that further qualify it (e.g., *web*). Like many modern information extraction approaches, we rely on external web resources, in our case Wikipedia and Stack Overflow. The approach is completely implemented and evaluated through a comparison with other available taxonomy tools [10].

In this research demonstration, we also introduce a web application for exploring the data produced by the Witt extraction procedure. The Witt Web Application is a new contribution, subsequent to the publication of the extraction technique. By supporting various data exploration scenarios (Section III), the Witt Web Application allows users to *(a)* better understand the challenges of hypernym detection in software engineering; *(b)* obtain structured information that describes a wide array of software technologies, and reason about how this information could be further integrated in other tools; and *(c)* obtain systematically-derived lists of all the software technologies within a category.

II. The Witt Tool Suite

We first present a summary of the Witt extraction tool, which automatically detects hypernyms from terms presumed to be software technologies, and aggregates these terms into a general list of categories. The description is necessarily concise, but a separate publication provides the complete details of all algorithms involved [10]. We then present an overview of the Witt Web Application, which complements the extraction technique.

The Witt *Extraction Tool*

As a vocabulary seed for software technologies, we considered the set of all Stack Overflow tags. Numbering over 50 000, these tags constitute the set of *canonical terms* for software technologies. This *canonical vocabulary* can then be expanded through synonyms and other equivalence relations (e.g., acronyms, contractions; see below).

The Witt extraction tool follows a three-step batch execution process. Once launched, it builds a *term database* that contains the category (hypernym) information for all terms in the canonical vocabulary. The term database can then be accessed and queried like any other database. The following paragraphs briefly describe each step.

The first step to create the term database is to obtain the *excerpt* and *information page* of Stack Overflow tags downloaded from the Stack Exchange API [11]. The tag excerpt is a short, unformatted summary of the tag, and the information page is a more complete, html-formatted, documentation page. The tag information is user-generated and can be missing.

The second step is to automatically identify the Wikipedia article (or article section) that describes the tag, if it exists. Despite the fact that Wikipedia has a search API [12], identifying the article that matches a Stack Overflow tag is far from trivial. "Some tags closely match a Wikipedia title, but for a different sense (e.g., the default article for ant refers to the insect, not the build tool). Other tags, such as curl, could reasonably be linked to more than one computer science related article. Also, some tags are only described within a section of a related article. For example, the tag catalina is described in the section Catalina of the article Apache Tomcat. These three challenges are compounded by the fact that we cannot assume that there is a Wikipedia article or article section for every tag" [10, Sect. 4]. To tackle these challenges, we designed a score to represent the similarity between a tag and an article, based on the text of the tag description and article and on the article's metadata.

In a third step, we apply a special-purpose hypernym detection algorithm to the article that describes the tag. The algorithm involves combining the output of four complementary detectors for identifying hypernyms for a term. Each detector can return zero, one, or multiple hypernyms. One detector analyses *Wikilinks* (hyperlinks in Wikipedia pages), and a second analyses *Infoboxes* (structured dictionaries for articles on certain topics). The two other detectors apply natural language processing techniques (NLP) to the text of the Wikipedia article, but also to the tag excerpt. One NLP detector is based on the concept of phrasal groups, whereas the other leverages grammatical relations.

In a final step, we transform the hypernyms detected in the previous step into a set of *categories* with *attributes*, while retaining the relation between a tag and its categories and attributes. This last step consists of a number of empirically-derived heuristics applied to the hypernyms to tokenize them, handle acronyms and compound terms, and then analyze word order and the use of prepositions and articles to distinguish categories from attributes. The output stored in the term database is a set of category–tag pairs, each associated with a set of attributes.

For example, Table I shows the distinction between the raw hypernyms and the corresponding category-attribute structure we generate. This data both illustrates the output of the tool at various stages, and motivates the need for the categorization stage. For example, the only raw hypernym shared by tags asp.net-mvc and django is software, hardly an insightful piece of information. The issue with simple hypernym extraction is that the extracted hypernyms are either too general or too specific. With the category structure in place, we can automatically determine that both tags are members of the category framework with the attribute web-application.

The Witt *Web Application*

The Witt Web Application acts as a portal to view and explore the contents of the term database, and enhances this content in several ways. The application has two basic modes of operation that correspond to the two directions of the hypernym/hyponym relation. Specifically, users can view the classification for a term (hypernymy mode), or obtain the list of terms in a category (hyponymy mode).

In hypernymy mode, the application enhances the content of the database by adding a tag searching feature and by resolving synonyms, using the existing synonymy relations in Stack Overflow. In hyponymy mode, the application can retrieve popularity data for different tags and categories directly from Stack Overflow. Popularity metrics can be computed in different ways: the current prototype counts the number of Stack Overflow posts tagged with a given tag.

TABLE I
SAMPLE HYPERNYM DETECTION AND CATEGORIZATION OUTPUT

asp.net-mvc	
Hypernym	open source web application framework open source web application framework and tooling software web application framework
Category: Attribute	framework: web-application, open-source software: tooling:
django	
Hypernym	free and open-source software free and open-source web framework free web framework open source server-side web application framework open-source web framework software web framework
Category: Attribute	framework: open-source, server-side, web-application software: free, open-source web-framework: free, open-source

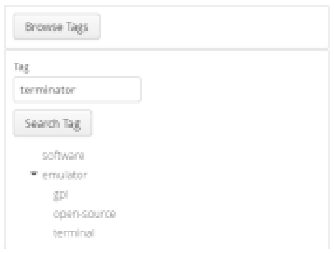

Fig. 1. Categories and attributes returned by the Witt Web Application when searching for the technology terminator.

III. Usage Scenarios

In this section, we describe three usage scenarios of the Witt Web Application.

What is This Technology?

To retrieve the classification for a given term (i.e., hypernymy mode), a user enters their query in the "Search Tag" search box. Figure 1 shows the results returned for the query terminator: The categories software and emulator, with the attributes gpl, open-source, and terminal for the latter.

The tag terminator illustrates some of the challenges that Witt has to overcome to retrieve the correct Wikipedia article: For the search term terminator, Wikipedia returns a disambiguation page which links to more than 50 articles, including entries from astronomy, genetics, and a series of science fiction films. Witt then calculates the likelihood of each article to be related to programming [10] and correctly identifies the article titled "Terminator (terminal emulator)" as the relevant one, extracting the categories and attributes from the article's leading sentence: "Terminator is an open-source terminal emulator programmed in Java."

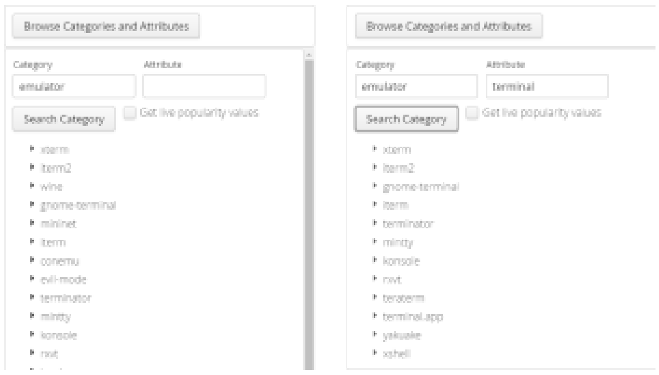

Fig. 2. Tags returned by the Witt Web Application that belong to the category emulator. On the right hand side, search results are limited to tags associated with the attribute terminal.

Exploring a Category

To explore the technologies contained in a category (i.e., hyponymy mode), a user specifies the category in the "Search Category" search box. The left hand side of Figure 2 shows all tags returned by the Witt Web Application for the category emulator, including the tag for the previously mentioned terminal emulator terminator and evil-mode. For the latter, Witt does not find a corresponding Wikipedia article or article section, but uses the Stack Overflow tag description instead for determining that evil-mode is an emulator: "Evil (Extensible Vi Layer for Emacs) is a vim emulator for emacs".

Finding a Technology with Specific Properties

To find a specific technology within the previously explored category, the user filters the tags found in a category by their attributes. The right hand side of Figure 2 shows the set of tags in the category emulator which are associated with the attribute terminal. The result still contains terminator, but no longer evil-mode. To distinguish the attribute terminal from the category emulator, Witt uses a number of empirically-derived heuristics based on all hypernyms retrieved for Stack Overflow tags.

IV. Evidence of Effectiveness

We evaluated the performance of Witt as part of our previous work on the development of the approach [10]. In this section, we summarize the main aspects of the evaluation method and results that can best inform how the Witt Web Application can be used.

Because there is no oracle or ground truth for either the set of all software technologies, the set of all categories of software technologies, or even sets of valid hypernyms/hyponyms for software technologies, we evaluated Witt by comparing it with state of the art taxonomy tools configured to expect technology terms.

As part of our evaluation, we compared the output of Witt with that of six comparable tools that were publicly available: *WebIsADb* [4], *WordNet* [3], *DBpedia Spotlight* [13], *WiBiTaxonomy* [14] *THD* [15], and *Google*'s "definition" feature. Because the comparison tools are not domain-specific, we injected additional information to contextualize the query to the programming domain. We manually verified that this made the tools perform better, resulting in a fairer comparison. For the evaluation, we considered three variants of our approach. One variant returns only the raw hypernyms (H). Another variant returns only the names of the general categories (C). The third variant returns both the categories and all their attached attributes (CA).

Our comparative evaluation mainly focused on measuring two aspects: the proportion of terms for which at least one *correct* hypernym is found, and the average number of wrong hypernyms returned by the tool for a term.

As a data set of input terms, we considered all Stack Overflow tags available at the time of the experiment (January 2018). Because the evaluation requires manual assessment of the results, we sampled a subset of the tags using a stratified sampling strategy of popular, common, and rare tags based on the number of mentions of each tag on Stack Overflow. We selected our samples for the three strata so that ratios observed for the subsample would have a confidence interval of 5% at the 0.95 confidence level.

We then input all the tags in the sample to each tool, and provided the aggregated resulting hypernyms to two of the authors for independent assessment, but without revealing the source of the hypernym to avoid investigator bias.

Figure 3, reproduced from our previous report [10], summarizes the overall results for coverage and average number of false positives by plotting the performance of the tools in two dimensions. Each data point represents the performance of one tool by combining the results for the three strata (popular, common, and rare terms) with a linear extrapolation that takes into account their relative cardinality. As the figure shows, the variant of Witt that returns raw hypernyms has the best coverage, but returns comparatively more spurious results than other alternatives. The tool with the lowest number of false positives is Google, but at the cost of very low coverage. When considering both dimensions (having equal weight), two of the variants of Witt offer the optimal solution to the problem of hypernym discovery. Our complete report [10] also includes an evaluation of the aggregating power of the categorization, which is a third important aspect of the approach not captured by this evaluation.

V. Conclusion

We introduced Witt, a tool suite to automatically classify software technologies. The Witt Web Application provides public access to a software technology taxonomy that can be systematically and continually updated by the Witt extraction process, and either queried by end users or integrated into other software engineering analytics applications.

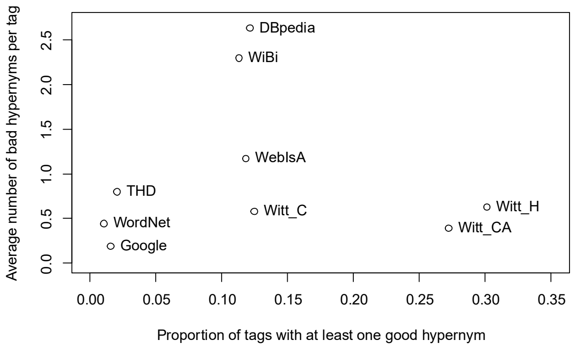

Fig. 3. Extrapolated aggregated performance across all strata. The bottom right corner indicates the overall best performance. Reproduced from [10].

References

[1] S. J. Gould, *Wonderful Life: The Burgess Shale and the Nature of History*. W. W. Norton & Company, 1989.

[2] E. Raymond, "The cathedral and the bazaar," *Knowledge, Technology & Policy*, vol. 12, no. 3, pp. 23–49, 1999.

[3] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller, "Introduction to Wordnet: An on-line lexical database," *International Journal of Lexicography*, vol. 3, no. 4, pp. 235–244, 1990.

[4] J. Seitner, C. Bizer, K. Eckert, S. Faralli, R. Meusel, H. Paulheim, and S. P. Ponzetto, "A large database of hypernymy relations extracted from the web," in *Proceedings of the 10th edition of the Language Resources and Evaluation Conference*, 2016, pp. 360–367.

[5] J.-R. Falleri, M. Huchard, M. Lafourcade, C. Nebut, V. Prince, and M. Dao, "Automatic Extraction of a WordNet-Like Identifier Network from Software," in *18th IEEE International Conference on Program Comprehension*, 2010, pp. 4–13.

[6] J. Yang and L. Tan, "SWordNet: Inferring semantically related words from software context," *Empirical Software Engineering*, vol. 19, no. 6, pp. 1856–1886, 2014.

[7] "Stack Overflow tags," Web Site, https://stackoverflow.com/tags, verified 17 Sep 2018.

[8] "Wikipedia:disambiguation," Wiki Page, https://en.wikipedia.org/wiki/Wikipedia:Disambiguation, verified 17 Sep 2018.

[9] "Comparison of web frameworks," Wiki Page, https://en.wikipedia.org/wiki/Comparison_of_web_frameworks, verified 17 Sep 2018.

[10] M. Nassif, C. Treude, and M. P. Robillard, "Automatically categorizing software technologies," *IEEE Transactions on Software Engineering*, 2018, accepted for publication. Available from https://doi.org/10.1109/TSE.2018.2836450.

[11] "Stack exchange API," REST API, https://api.stackexchange.com/, verified 17 Sep 2018.

[12] "Mediawiki action API," REST API, https://www.mediawiki.org/wiki/API:Main_page, verified 17 Sep 2018.

[13] P. N. Mendes, M. Jakob, A. García-Silva, and C. Bizer, "DBpedia Spotlight: Shedding light on the web of documents," in *Proceedings of the 7th International Conference on Semantic Systems*, 2011, pp. 1–8.

[14] T. Flati, D. Vannella, T. Pasini, and R. Navigli, "Two is bigger (and better) than one: the Wikipedia Bitaxonomy Project," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, 2014.

[15] M. Dojchinovski and T. Kliegr, "Entityclassifier.eu: Real-time classification of entities in text with wikipedia," in *Machine Learning and Knowledge Discovery in Databases*, ser. Lecture Notes in Computer Science, H. Blockeel, K. Kersting, S. Nijssen, and F. Železný, Eds. Springer, 2013, vol. 8190, pp. 654–658.