Semi-Supervised Wikipedia Category Suggestion

Benjamin Perez
bperez@seas.upenn.edu
Univ. of Pennsylvania
Philadelphia, PA

Cristoforo Feo
cfeo@seas.upenn.edu
Univ. of Pennsylvania
Philadelphia, PA

Andrew G. West
westand@cis.upenn.edu
Univ. of Pennsylvania
Philadelphia, PA

Insup Lee
lee@cis.upenn.edu
Univ. of Pennsylvania
Philadelphia, PA

ABSTRACT

In 2005 Wikipedia implemented a category system for the purposes of facilitating navigation throughout the site. Since then, it has grown to 1.5 million categories covering over four million articles. Currently, all categorization of articles on Wikipedia is done by human editors—a task which involves an enormous amount of repetitive work.

This paper proposes building an automated system that suggests missing category-article links to editors through a browser-based user-interface. This system will utilize the existing corpus of human-categorized articles, as well as available metadata, to provide a ranked list of suggested categorizations. All suggested categorizations are subject to final approval or rejection by a human editor.

1. INTRODUCTION

Fundamentally, Wikipedia consists of a number of content pages, or articles, and a series of categories to which articles might be linked. Categories may have sub categories and pages may belong to several different categories. In a traditional knowledge base this format works well for classifying many different types of objects. If a new object is to be added, the editors will follow whatever criteria they have laid out for placing items into their correct categories. However, the unique nature of Wikipedia as a massively collaborative resource presents an interesting challenge for categorization [15]. In particular, all editing is performed by volunteers who are wildly diverse in terms of geography, education, and professional interests. While this is great for the collection of knowledge in general, it is not ideal for categorization—where there are many opportunities for misunderstanding [23]. Figure 1 shows a subset of the complex web of categories and sub-categories applied to the Wikipedia article on George Washington.

Take, for example, the Wikipedia page for "George Washington". Suppose an editor writes this article and categorizes it under "Presidents of the United States". Later, another author may write an article on "James Monroe" and decide to create a new category for "People from Westmoreland County, Virginia". Unless the author of the "George Washington" article later visits the "Thomas Jefferson" page, she will not know that the "Westmoreland County" category even exists and so it will be missing from the "George Washington" page . These kinds of missing links exist in many places on Wikipedia and are incredibly difficult to detect through casual browsing. The category system itself suffers from a number of ambiguities that make this a difficult problem to solve. Perhaps the greatest is a fundamental disagreement over how categorizations should work first identified by Thornton [22]. She noted that the two ways of looking at categories, hierarchical and relational, are both currently used by Wikipedia. This makes automated category suggestion relatively difficult with traditional machine learning techniques that attempt to extract a single type of semantic meaning from article categorizations.

In spite of this, it is clear that the category system serves its purpose as a navigational tool tying together articles that share some measure of similarity. Therefore, we plan to utilize an approach that leverages the existing pseudo-hierarchical category structure of Wikipedia as a guide. In fact this is exactly what the recommended procedure for categorizing new articles suggests: "One way to determine if suitable categories already exist for a particular page is to check the categories of pages concerning similar or related topics." [25] It is our goal to create a tool which will find these missing category links and bring them to the attention of editors so that they might be quickly and easily added. This will benefit the category system as a whole and potentially simplify the categorization process for a wider audience of editors [7].

2. RELATED WORK

Because Wikipedia is such a rich, diverse corpus of information, a lot of areas of machine learning research use Wikipedia as a training set for a variety of learned models. In this way, Wikipedia is used to improve the quality of machine learning research. This paper proposes the opposite: to use machine learning techniques to improve the quality of Wikipedia.

Automated classification of Wikipedia categories is an area that has been studied from several different angles. The category system contains a huge amount of extractable information, although it is very loosely structured [13, 18]. This loose structure arises naturally in Wikipedia, since articles are written and edited collaboratively. Furthermore, category links are not applied in a consistent or systematic manner across all of Wikipedia. The most simplistic approaches to categorizing Wikipedia utilize supervised learning. Supervised learning takes as input a set of training examples and their associated labels. In our case, these training ex-

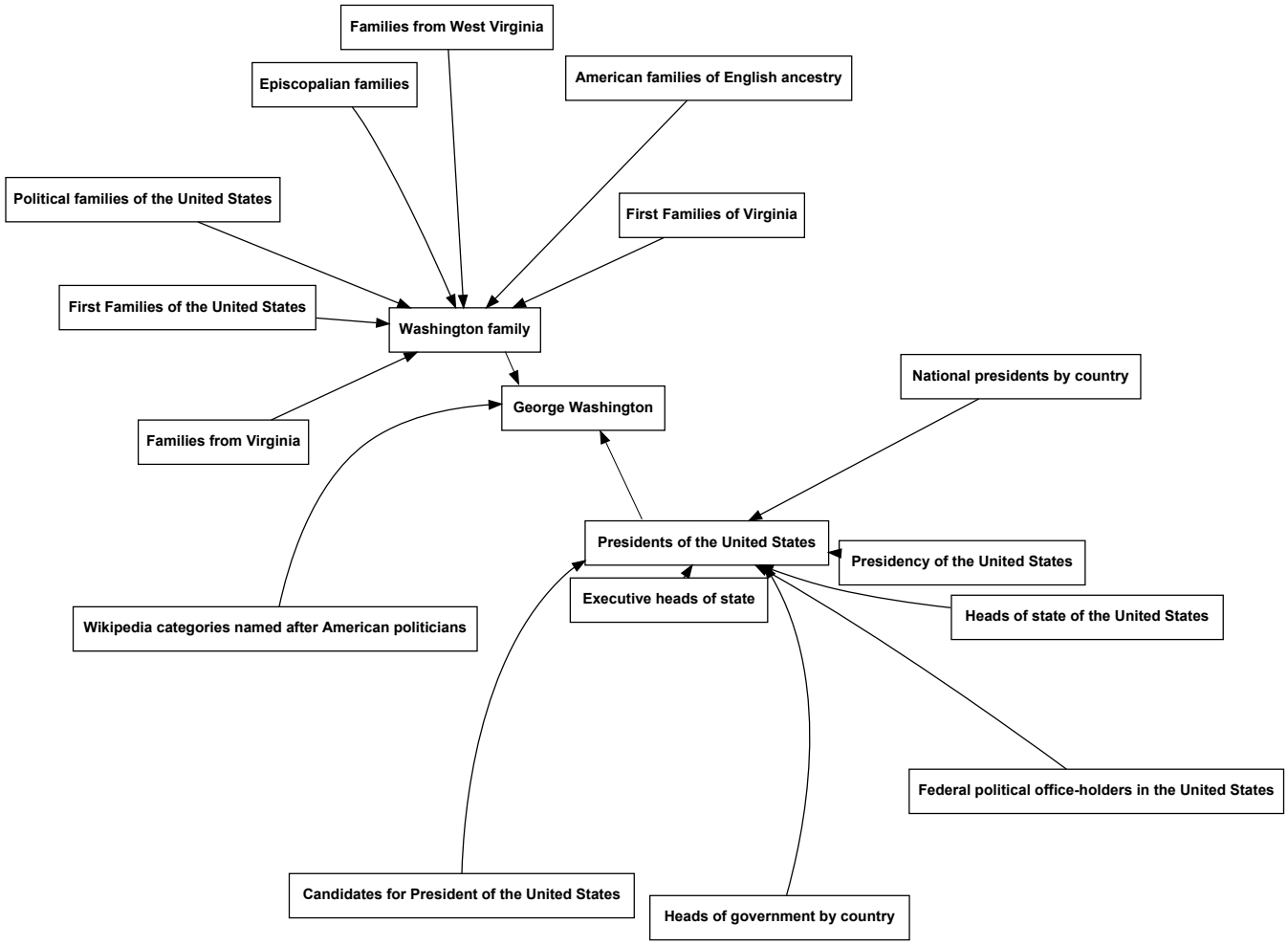

Figure 1: A small part of the category heirarchy for the Wikipedia article on "George Washington".

amples are Wikipedia articles and their labels are the categories associated with them. Supervised learning uses these labeled training examples to learn a model which can predict the labels of previously unseen examples [21].

There are two broad groups of features which are used as input to supervised learning algorithms: content features and network features. Content features reflect information directly contained within a given page. Examples of content features include textual keywords, infobox information, and metadata associated with a specific article [24]. Network features describe the hyperlinked graph of Wikipedia articles and may contain information such as the number of outgoing links, the number of incoming links, and properties of neighbors in this hyperlinked graph [16].

The use of content features or network features alone has yielded impressive results [8, 21]. Using a combination of both feature types has been shown to be even more effective [8]. Although supervised learning techniques are highly accurate for a few very specific categories [8, 7, 21, 24], they suffer from problems of scale. This is primarily because a classifier must be trained for each and every category. This approach will clearly not scale across the entire category system of Wikipedia, with its large and constantly changing set of categories [7]. Furthermore, previous work demonstrated that supervised learning is best for distinguishing between a few very distinct categories. This assumption does not hold across the vast majority of the category system, where categories may overlap or contain subtle differences [22]. All of these limitations arise because supervised learning works primarily at the level of individually labeled articles, fundamentally limiting the amount of information that can be incorporated from the entire Wikipedia graph, even when network features are included.

Another approach to categorizing Wikipedia is to use semi-supervised learning algorithms. Semi-supervised learning differs from supervised learning in that it uses unlabeled data in the learning process. Many of these methods rely on label-propagation across a graph. The input to a semi-supervised learning algorithm is generally a set of examples. This set of examples is divided into labeled and unlabeled data. Using the assumption that the data are distributed in a way which correlates with their labels, the missing labels in the data can be estimated [5].

Azran [3] describes a generic example of an algorithm—known as the rendezvous algorithm—which can estimate a multi-class distribution of labels across examples. The rendezvous algorithm works by treating each example as a node in a fully connected graph. Each edge in this graph

has an associated weight which is given by its entry in a pairwise similarity matrix between all examples called the transition matrix. Labeled example nodes are set to be absorbing states and unlabeled example nodes are set to be emitting states in a Markov random walk. This means that each unlabeled example will emit a particle that chooses a neighbor to move to based on the probabilities defined in the transition matrix. At each new node, this particle chooses its next step based only on the probabilities defined for its current location—the "Markov" aspect of the random walk. Upon reaching a labeled node, the particle is absorbed and the label of the node is recorded. After running a number of random walks for each unlabeled node, the distribution of labels for that node can be inferred by looking at the distribution of absorbing states for its random walks.

Chidlovskii describes the results of a related random walk algorithm on Wikipedia. One of the important concerns he raises is the difficulty of computing a pairwise similarity matrix for a dataset as large as Wikipedia. One way to avoid this problem is to use the graph of hyperlinks between pages as a starting point for the transition matrix. Although the specifics of Chidlovskii's algorithm differ from the rendezvous algorithm, they demonstrate that label-propagation approaches have enormous potential in sparse graph applications such as Wikipedia [6].

Graph based approaches such as these avoid the scalability bottleneck of needing to train an individual classifier for each category on Wikipedia. Furthermore, they allow us to incorporate all of the information inherent in the hyperlink structure of Wikipedia in our inference [2]. This link structure between articles has been found to be very highly correlated with the link structure of categories [13, 10], leading us to believe that it will be quite useful.

This paper plans to build on the preliminary results described here to develop a label propagation technique which can scale to the entire Wikipedia dataset. This approach will combine the robust, multiclass nature of a technique like the rendezvous algorithm with approaches for dealing with scalability similar to those Chidlovskii describes to produce a superior classification system.

3. SYSTEM MODEL

Our approach to this problem is twofold: first prune the input data to a more reasonable subset of potential article-category pairings. Then apply machine learning techniques using features inherent in each pairing to create a decision tree or other classifier which outputs a confidence in the correctness of the missing link. We then plan to integrate these results into a browser-based plugin which can provide a ranked list of the potential categorizations to editors as they browse through Wikipedia.

We will first generate rough suggestions by running a graph-based "first-pass" algorithm, which will provide potential missing categorizations in a graph of articles. The final result should exhibit high recall while still significantly reducing the number of possible categorizations for an article [2]. These potential article-category pairings will be used later as a basis for feature selection.

Next we take a statistically sampled subset of these potential categorizations and have individuals—ideally Wikipedia editors themselves, if we can recruit them—label each article-category pair as either good or bad. We plan to use the machine learning software Weka [9] to create a model for scoring and ranking the suggestions we have obtained from our graph-based approach. While high-recall was the goal of the graph-based algorithms, here we want to improve the precision of the suggestions we will ultimately produce.

One important factor in learning such a model is determining the features that go into the algorithm. One feature that we will input to the algorithm is the numerical score from the graph algorithm. Other features that we will explore will include document similarity measures between an article and articles with the suggested category, metadata about the article and suggested category, and possibly properties of the category graph such as subcategory or supercategory relationships.

Finally if time allows and our techniques are robust enough we plan to make our data available to editors by integrating with the popular tool HotCat [12]—which eases category management for Wikipedians. It is our plan to offer a simple ranked list of the top five or so potential categorizations for an article such that an editor visiting a page can quickly glance at our tool and check whether any of the supplied categories are appropriate. Figure 2 illustrates the proposed architecture of the entire system in block-diagram form.

4. SYSTEM IMPLEMENTATION

Thus far we have completed implementation of the graph-based first phase. We chose Java as our primary language due to its acceptable performance for general computing tasks as well as our past experience using Java in large projects. Another consideration in choosing Java is the large number of libraries available as well as the possibility of using a distributed computing framework like Hadoop in the future.

Initially we planned to represent Wikipedia articles and categories as a graph-structure which could be contained entirely in memory. However, because all of the data we need is available as database tables this was unnecessary. These tables are available as SQL and can be obtained directly from Wikipedia. We chose to implement a system which uses calls to a mySQL database to lazy load articles and links as they are needed. For example, when our algorithm needs to find all outgoing links from an article, it queries a manager object which maintains a cache of all previously loaded pages and loads required pages on cache misses. This results in very little memory overhead for our running processes and a minimal number of queries to the database.

We have used this framework to build and test two different types of graph algorithms. Both algorithms take as input a specific article and return a list of potential new categories. One of the algorithms also outputs a score for each new category. These scores roughly correspond to the categories' relative frequency of occurrence in the input article's hyperlinked neighbors.

The first technique is a simple breadth-first search of all hyperlinked articles. This search returns the first n categories it finds, where n can be specified by the algorithm. Starting from a root article, the breadth-first search inspects the first adjacent (hyperlinked) article's categories and adds them to the output set of suggestions. This process is repeated in a breadth-first manner until the specified number of new categories are found. While this technique is fairly simple to implement, it does not output scores for each new category..

The second technique utilizes random walks to find and

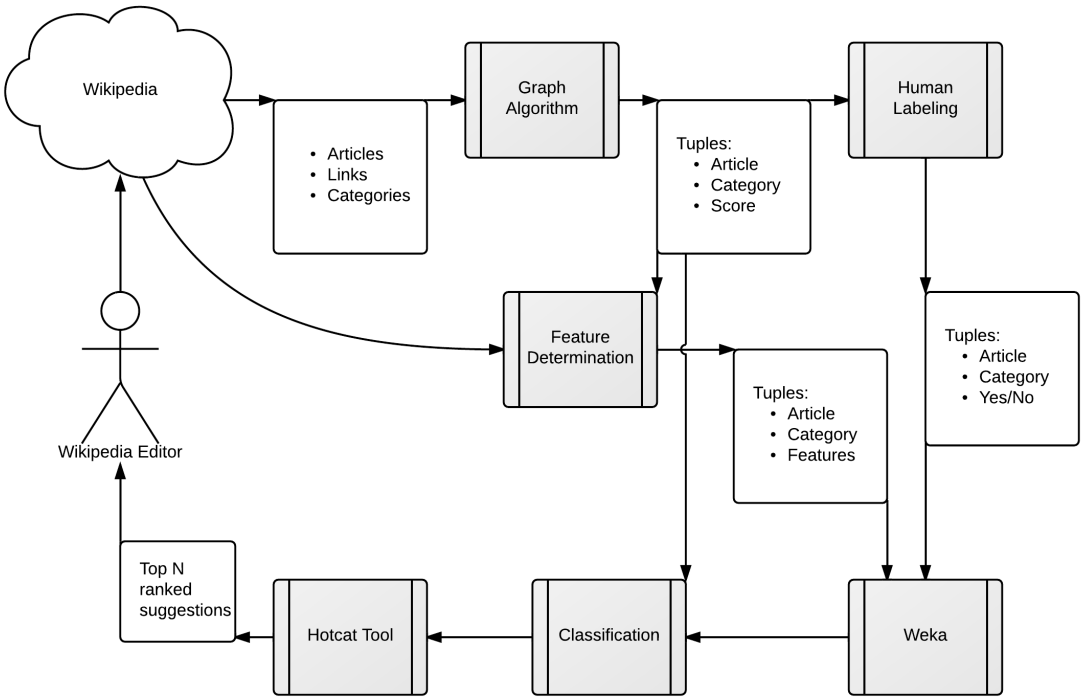

Figure 2: Our proposed system architecture. Gray blocks are processes, while arrows and white blocks represent data interchance between components.

return a weighted score for new category suggestions. Starting from a root article, the algorithm queries all of its hyperlinked neighbors. The random walk currently has an equal probability of jumping to any of these neighbors. For walks of length greater than one, this process is repeated multiple times for each article jumped to. When a random walk has completed a number of jumps equal to its length, it adds the categories of the article it ended on to the output set of suggestions for the root article.

This technique runs a fixed number of iterations for each article. Each iteration currently performs a walk of length one, two, and three. A count of all new categories returned at the end of each walk is maintained. In order to weight categories that are farther from the root node lower, counts are incremented at a rate inversely proportional to their distance from the article. For example, seeing a category at distance three is worth one third of the weight of a category at distance one. At the end of all the iterations each category is given a final weight consisting of its count divided by the sum of all category counts. The results are then sorted and returned as suggestions.

One issue we have encountered is the inclusion of many administrative pages and categories in our results. While Wikipedia has an easy method for distinguishing administrative articles from informational articles, no such method exists for categories. We have filtered out administrative articles by ignoring anything not in the default namespace. Administrative categories are more difficult to filter, as there is no explicit identifier for them in the database tables provided by Wikipedia. We have been examining heuristics that take advantage of the relatively straightforward Wikipedia naming conventions for administrative categories. For example, one can usually safely ignore categories which begin with "Articles_with", "Articles_lacking", "Articles_needing", or "All_articles_with".

5. SYSTEM PERFORMANCE

5.1 Desired Properties

Since our next milestone is to have annotators label a sample of good and bad category suggestions for articles, the desired properties of the of the graph algorithm are focused on efficiently generating a manageable list of recommendations. Therefore, the algorithm must have at least the following properties:

- High Recall: Because the output from this algorithm will later be filtered by a classifier or machine-learned

model, it is important that this phase of the project returns as many correct categories as possible. Poor categories can be filtered out later, but additional good suggestions cannot be generated after this stage.
- Scalable Performance: Wikipedia is a vast dataset containing millions of articles which need categorization. This graph-based portion of the project works directly on each of these articles and their neighbors, meaning it will likely be the bottleneck for the entire system. Our tests on individual categories or small groups of categories must be able to scale to all of Wikipedia.

Two additional, though not strictly required properties are listed below:

- Confidence Scores: A confidence score would be a valuable input to building a model to classify suggestions as good or bad. This would also probably expedite the labeling process, since only the most relevant suggestions could be presented to annotators for labeling.
- Removal of Irrelevant Results: As previously mentioned, removal of administrative pages and categories would reduce the number of pages requiring subsequent classification, as well as reduce noise in subsequent classification attempts. This could be performed in the classification phase of the project, but incorporating information from this graph-based approach may also prove helpful for removing irrelevant results.

With these properties in mind, we have collected some preliminary results to guide our future efforts as the project progresses.

5.2 Recall

Figure 1 displays example output—suggested categories—from the two algorithms for the article on Danish writer Hans Christian Andersen. While only the random walk technique provides explicit scores for each suggestions, we can see that the relevance of the suggestions from breadth-first search also decreases from top to bottom. Both techniques produce similarly relevant results. The relative simplicity of the breadth-first search technique is probably offset by the random walk technique's tendency to travel and collect categories from further and possibly less relevant pages. These categories tend to focus on locations and dates, particularly for pages on historic figures—which tend to be part of many list articles. The higher relevance of earlier results in the breadth-first method can be explained by the fact that categories which appear earlier are taken from pages which are located closer to the root article.

We also ran a test to determine the size of the intersection—or overlap—between the two methods output for a given article. We examined how many categorizations out of 100 were suggested by both algorithms on three different pages. The results are summarized in Figure 1 for these three articles. There was only about 10% overlap between the two algorithms.

5.3 Scalable Performance

The average running time of each algorithm on three different articles, as well as the overlap of their suggestions, is recorded in Figure 2. While the breadth-first algorithm is slightly faster in most cases, the two are close enough that the difference is largely irrelevant. More interesting, the runtimes are all around one second. This means that for the four million content pages on wikipedia it would take about 46 days to generate 100 suggestions for all pages. In practice this number will likely be lower, since the algorithm will likely be running on faster hardware. However, the runtime could still benefit from optimization and this is an area we are looking into moving forward.

6. REMAINING WORK

Our immediate next step is to develop more extensive heuristics to filter out administrative categories. This is a matter of determining a wider array of title types for administrative categories. Following this, our next priority is getting sample data from the graph algorithm labeled. We are currently exploring options for achieving this, including Mechanical Turk and Wikipedia editors. By late February, our plan is to have obtained the labeled data and to begin using Weka to train a classifier for further scoring category suggestions. Finally, we will use the results from our classifier to further refine suggestions and ultimately integrate with HotCat. If there is time, we would still like to collect feedback from editors to assess the usefulness of the suggestions. Thus far we have kept up with our project timeline—as laid out in our proposal—and have completed about 45% of the project to date.

7. REFERENCES

[1] Sisay Fissaha Adafre and Maarten de Rijke. Discovering missing links in wikipedia. In *Proceedings of the 3rd international workshop on Link discovery*, LinkKDD '05, pages 90–97, New York, NY, USA, 2005. ACM.

[2] Konstantin Avrachenkov, Paulo Gonçalves, Alexey Mishenin, and Marina Sokol. Generalized optimization framework for graph-based semi-supervised learning. *CoRR*, abs/1110.4278, 2011.

[3] Arik Azran. The rendezvous algorithm: multiclass semi-supervised learning with markov random walks. In *Proceedings of the 24th international conference on Machine learning*, ICML '07, pages 49–56, New York, NY, USA, 2007. ACM.

[4] Sharon Ann Caraballo. *Automatic construction of a hypernym-labeled noun hierarchy from text*. PhD thesis, Providence, RI, USA, 2001. AAI3006696.

[5] Andrew Carlson, Justin Betteridge, Estevam R. Hruschka, Jr., and Tom M. Mitchell. Coupling semi-supervised learning of categories and relations. In *Proceedings of the NAACL HLT 2009 Workshop on Semi-Supervised Learning for Natural Language Processing*, SemiSupLearn '09, pages 1–9, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[6] Boris Chidlovskii. Advances in focused retrieval. chapter Semi-supervised Categorization of Wikipedia Collection by Label Expansion, pages 412–419. Springer-Verlag, Berlin, Heidelberg, 2009.

[7] Linyun Fu, Haofen Wang, Haiping Zhu, Huajie Zhang, Yang Wang, and Yong Yu. Making more wikipedians: facilitating semantics reuse for wikipedia authoring. In *Proceedings of the 6th international The semantic web*

Hans Christian Anderson	
Random Walk	Breadth-First-Search
port_cities_and_towns_of_the_baltic_sea	knights_of_the_golden_fleece
academy_honorary_award_recipients	denmark_norway
translators_from_danish	history_of_santa_barbara_county,_california
cecil_b_demille_award_golden_globe_winners	use_dmy_dates_from_august_2011
copenhagen	1768_births
municipal_seats_in_capital_region_of_denmark	danish_american_history
danish_silent_film_actors	extra_knights_companion_of_the_garter
use_dmy_dates_from_august_2012	regions_of_italy
american_people_of_danish_descent	characters_in_fairy_tales
translators_to_english	dukes_of_schleswig
knights_of_the_order_of_the_dannebrog	norwegian_monarchs
articles_containing_potentially_dated_statements_from_january_2012	danish_migration_to_north_america
presidents_of_the_academy_of_motion_picture_arts_and_sciences	grand_commanders_of_the_order_of_the_dannebrog
burials_at_forest_lawn_memorial_park_(glendale)	calabria
articles_containing_potentially_dated_statements_from_july_2012	cities_in_santa_barbara_county,_california
articles_containing_danish_language_text	dukes_of_saxe-lauenburg
articles_containing_potentially_dated_statements_from_2011	works_by_hans_christian_andersen
municipal_seats_of_denmark	bruttium
danish_translators	1839_deaths
european_capitals_of_culture	use_dmy_dates_from_august_2012
capitals_in_europe	regents
port_cities_and_towns_in_denmark	burials_at_roskilde_cathedral
danish_film_actors	people_from_copenhagen
hans_christian_andersen	peninsulas_of_italy
populated_places_established_in_the_11th_century	19th-century_monarchs_in_europe
cancer_deaths_in_california	fairy_tales
danish_emigrants_to_the_united_states	danish_monarchs
all_articles_containing_potentially_dated_statements	wine_regions_of_italy
1956_deaths	protestant_monarchs
1886_births	incorporated_cities_and_towns_in_california
cities_and_towns_in_capital_region_of_denmark	populated_places_established_in_1911
danish_dramatists_and_playwrights	literature_featuring_anthropomorphic_characters
norwegian_essayists	nuts_2_statistical_regions_of_the_european_union
danish_essayists	regents_of_denmark
denmark_norway	accuracy_disputes_from_august_2012
norwegian_dramatists_and_playwrights	fictional_birds
visitor_attractions_in_santa_barbara_county,_california	visitor_attractions_in_santa_barbara_county,_california
incorporated_cities_and_towns_in_california	house_of_oldenburg
18th-century_danish_people	1911_establishments_in_the_united_states
cities_in_santa_barbara_county,_california	persondata_templates_without_short_description_parameter

Table 1: Top 40 Suggested Categories From Each Algorithm

Article Title	Suggestion Overlap (%)	Breadth-First Runtime (s)	Random Walk Runtime (s)
Hans Christian Andersen	5.21544565418	1.00786779928	1.17027116942
Acoustic Theory	14.4684335914	1.06326193118	1.05459133649
Gun Control	10.77322996802	1.09626546693	1.2392341857

Table 2: Algorithm Runtimes and Suggestion Overlap

and 2nd Asian conference on Asian semantic web conference, ISWC'07/ASWC'07, pages 128–141, Berlin, Heidelberg, 2007. Springer-Verlag.

[8] Zeno Gantner and Lars Schmidt-Thieme. Automatic content-based categorization of wikipedia articles. In *Proceedings of the 2009 Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources*, People's Web '09, pages 32–37, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[9] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18, November 2009.

[10] Todd Holloway, Miran Bozicevic, and Katy Börner. Analyzing and visualizing the semantic coverage of

wikipedia and its authors: Research articles. *Complex.*, 12(3):30–40, January 2007.

[11] Glen Jeh and Jennifer Widom. Simrank: a measure of structural-context similarity. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '02, pages 538–543, New York, NY, USA, 2002. ACM.

[12] Magnus Manske. Hotcat.

[13] Simone Paolo Ponzetto and Michael Strube. Deriving a large scale taxonomy from wikipedia. In *Proceedings of the 22nd national conference on Artificial intelligence - Volume 2*, AAAI'07, pages 1440–1445. AAAI Press, 2007.

[14] Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th international joint conference on Artificial intelligence - Volume 1*, IJCAI'95, pages 448–453, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

[15] Matthew Richardson and Pedro Domingos. Building large knowledge bases by mass collaboration. In *Proceedings of the 2nd international conference on Knowledge capture*, K-CAP '03, pages 129–137, New York, NY, USA, 2003. ACM.

[16] Prithviraj Sen and Lise Getoor. Link-based classification. Technical Report CS-TR-4858, University of Maryland, February 2007.

[17] Rion Snow. Semantic taxonomy induction from heterogenous evidence. 2006.

[18] Michael Strube and Simone Paolo Ponzetto. Wikirelate! computing semantic relatedness using wikipedia. In *proceedings of the 21st national conference on Artificial intelligence - Volume 2*, AAAI'06, pages 1419–1424. AAAI Press, 2006.

[19] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: A core of semantic knowledge - unifying WordNet and Wikipedia. 2007.

[20] Omer Sunercan and Aysenur Birturk. Wikipedia missing link discovery: A comparative study, 2010.

[21] Julian Szymański. Towards automatic classification of wikipedia content. In *Proceedings of the 11th international conference on Intelligent data engineering and automated learning*, IDEAL'10, pages 102–109, Berlin, Heidelberg, 2010. Springer-Verlag.

[22] Katherine Thornton. Contentious categories: Discussions of the design of the category system in wikipedia. In *North American Symposium on Knowledge Organization*, 2011.

[23] Katherine Thornton and David W. McDonald. Tagging wikipedia: Collaboratively creating a category system. In *Proceedings of the 2012 ACM Conference on Supporting Group Work*, 2012.

[24] Maksim Tkachenko, Alexander Ulanov, and Andrey Simanovsky. Fine grained classification of named entities in wikipedia. Technical report, HP Laboratories, 2010.

[25] Wikipedia. Wikipedia: Categorization. `http://en.wikipedia.org/wiki/Wikipedia:Categorization#Categorizing_pages`.