A Support Tool for Deriving Domain Taxonomies from Wikipedia

Lili Kotlerman^{*}, Zemer Avital[§], Ido Dagan^{*}, Amnon Lotan[‡], Ofer Weintraub[§]

*Bar-Ilan University, [§]Orca Interactive Ltd., [#]Tel Aviv University

lili.dav@gmail.com, zemer.avital@orcainteractive.com, dagan@cs.biu.ac.il, amnonlot@post.tau.ac.il, ofer.weintraub@orcainteractive.com

Abstract

Organizing data into category hierarchies (taxonomies) is useful for content discovery, search, exploration and analysis. In industrial settings targeted taxonomies for specific domains are mostly created manually, typically by domain experts, which is time consuming and requires a high level of expertise. This paper presents an algorithm and an implemented interactive system for automatically generating target-domain taxonomies based on the Wikipedia Category Hierarchy. The system also enables human post-editing, facilitated by intelligent assistance.

1 Introduction

Hierarchies of category names (taxonomies) are very useful for effective information access (Käki (2005), Stoica et al. (2007)). When geared for a specific domain or data collection, such hierarchies can highly benefit the tasks of content discovery, search, exploration and analysis. Our project, carried out by the Natural Language Processing group at Bar-Ilan University and Orca Interactive Ltd., aimed at semi-automatic generation of a taxonomy for the domain of general video content in order to enhance search and improve recommendations in a personalized video recommendation system.

This paper delivers two main contributions: (1) a novel algorithm for automatic generation of target-domain taxonomies and (2) an interactive taxonomy editing tool, which helps human editor to post-edit and improve automatically generated taxonomies by providing her with intelligent assistance.

Automatic taxonomy generation approaches can be roughly divided into two classes: corpus-based and knowledge-based. We suggest a knowledgebased algorithm, deriving focused target-domain taxonomies from the Wikipedia Category Hierarchy (WCH). WCH covers a very wide range of topics and is assumed to embed smaller taxonomies suitable for specific domains. The algorithm is thus aimed at extracting such taxonomies from WCH.

Since automatic techniques for taxonomy creation are not accurate enough, in real-life applications some human post-editing is usually employed. Our taxonomy editing tool was designed to facilitate this process. It provides the editor with intelligent assistance, based on statistical similarity in a domain corpus along with WCH. Our initial experiments in the video domain show considerable reduction of time needed for taxonomy generation, as well as improvement of the taxonomy quality, compared to a manually created taxonomy.

In Section 2 we describe some prior art and essential background. Section 3 describes our suggested taxonomy generation algorithm, while Section 4 describes the taxonomy editing tool.

2 Background

2.1 Taxonomy Generation

Two major approaches to automatic domain taxonomy generation can be identified in the literature. The first is the corpus-based approach, in which hierarchical clustering methods are applied either directly to keyword terms extracted from a targetdomain corpus for generating a keyword hierarchy, or to the documents in the corpus with further extraction of category names as keywords frequent in each cluster¹. These methods consider distributional corpus statistics and reflect the actual trends in the data, yet the resulting hierarchies are rather noisy and category names are not easily understandable.

¹See a summary at (Krishnapuram and Kummamuru, 2003)

The second, knowledge-based approach relies on manually constructed lexical hierarchies, such as WordNet (Fellbaum, 1998). For example, the Castanet algorithm (Stoica et al., 2007) utilizes *is-a* relations within WordNet to organize keywords into a hierarchy. Such hierarchies are more accurate than those obtained by clustering. Some related studies that compare clustering with knowledge-based category systems show that participants prefer categories (Pratt et al., 1999). The disadvantage of such hierarchies is their limited coverage.

In our work we follow the knowledge-based approach. We suggest utilizing the most comprehensive category hierarchy available, namely Wikipedia Category Hierarchy, in order to obtain relatively accurate taxonomies and avoid the disadvantage of limited coverage. In addition, we combine distributional information to better reflect the actual trends in the data, similarly to corpus-based methods.

2.2 Wikipedia Categories

The majority of Wikipedia articles, each usually describing a single topic, have been manually assigned to one or multiple categories. These categories are arranged in a hierarchy, which we refer to as Wikipedia Category Hierarchy (WCH). WCH is widely used for research, including generation of large-scale taxonomies and ontologies (de Melo and Weikum (2010), Ponzetto and Navigli (2009), Ponzetto and Strube (2007), Suchanek et al. (2007)). Yet, to the best of our knowledge, WCH was never previously used to address our task of creating focused target-domain taxonomies.

The main advantages of WCH are that it is multilingual, covers almost all conceivable topics and is constantly evolving, thus never going out of date. WCH has a single root node. Deeper-level categories have many subcategories and parent categories, while Wikipedia articles are placed at the leaves of the hierarchy. Thus, the hierarchy approximates a directed acyclic graph (DAG). In our work to obtain a strict DAG we performed a preprocessing step that removed the few cycles existing in WCH. Figure 1 presents an excerpt of WCH for ancestors of the *surfing* node.

2.3 Distributional Similarity

To derive target-domain taxonomies from WCH, our algorithm utilizes distributional similarity



Figure 1: Paths from the *surfing* node to the root of WCH. The edges are directed from subcategories towards parent categories.

scores between category names. The distributional similarity approach assumes that terms that appear with similar context words have similar meanings.

We suggest that if similarity scores are calculated based on a corpus representing a target domain, then terms distributionally similar to a given category name c indicate the typical context or sense of c in the given domain. For example, in the recipes domain *cookie* will be similar to *biscuit*, while in texts about the Web *cookie* will be most similar to *file*.

In our work we used a directional distributional similarity measure (Kotlerman et al., 2010), which learns directional similarities between specific terms and more general ones, e.g. *koala* \rightarrow *animal*, *wedding* \rightarrow *marriage*. This type of similarity better corresponds to our task of building category hierarchies, in which the relations between category nodes are directional - from specific to more general ones.

3 Generating an Initial Domain Taxonomy

As explained in Section 2.2, WCH covers a very broad range of conceivable subjects and fields of interest and thus embeds various target-domain taxonomies. Accordingly, we define a targetdomain taxonomy as a subtree of WCH and determine our goal as deriving such a subtree from WCH. Our preliminary analysis within the video domain showed that indeed almost all the desirable domain categories were found in WCH.

We address our goal in three stages:

- 1. Detect *target categories* a subset of WCH categories relevant for the target domain.
- 2. Form an initial subtree by picking out for each

target category a single path to the root of WCH amongst all possible paths.

3. Prune the resulting hierarchy to retain only the most relevant categories and obtain a taxonomy of the desired size.

3.1 Detecting Target Categories

To define a relevant subset of WCH nodes we suggest using a set of keywords, including multi-word ones, representing the important concepts of the domain. Such keywords can be extracted from a corpus representing a target domain, which is a common practice for automatic taxonomy generation, or obtained from a target-domain collaborative tagging system. For our target video domain we used keywords obtained from the IMDb² collaborative tagging system, where users assign keywords to movie descriptions.

The keyword set is intersected with the set of WCH category names in order to obtain the list of target categories. Keywords not found in WCH are discarded. In our experiments most of the discarded keywords indeed were not valuable as category names e.g. *based on novel*, *young boy*. Others had a synonymous keyword found in WCH, e.g. *automobile accident* and *car accident*.

We note that it is worth using an exhaustive list of target categories, larger than the desired taxonomy size. Though not all of the target categories will be retained during pruning, each one will contribute when deciding on the importance of its parent category.

For each of the resulting target categories its *domain frequency* is specified, which can stand either for the number of corresponding keyword's occurrences in the target-domain documents or for the number of documents annotated with this keyword in a (manual) tagging system.

3.2 Deriving a Target Taxonomy Subtree

In Figure 2 we present the outline of our suggested algorithm, which given WCH and a set of target categories C as its input generates a target taxonomy tree T. As explained above, to generate the taxonomy tree a single path from each target category to the root of WCH should be chosen. We address this goal iteratively, by processing at each step one *current* target category c (step a) and selecting for it a single parent category p', based the on weighting heuristics we explain below (steps

Input:	
W - Wikipedia Category DAG	
C - list of target categories sorted by depth in W	
Output:	
T - taxonomy tree	
1.	Initialize T as an empty tree
2.	While C not empty do:
	a. Pop a category c from the head of the list C
	b. $P = \text{all parents of } c \text{ in } W$
	c. $p' = argmax_{p \in P}(weight(c, p))$
	d. Add edge $(c \rightarrow p')$ to the taxonomy tree T
	e. If p' not in $C \cup T$: add p' to C
3.	Prune T to remove marginal categories

Figure 2: Algorithm outline.

band c). If the selected parent category is not found in the list of target categories then it is added to the list to further proceed with the path construction (step e). We sort the input list of target categories by their depth in WCH, so that daughter categories would be processed before their parents.

Below we describe the method we suggest for selecting the most suitable parent p' for a given target category c. As can be seen from the outline, the highest-scoring parent is selected, while we suggest the weight assigned to each candidate parent p to be the product of two factors:

 $weight(c, p) = w_{self}(p) \cdot w_{daughter-parent}(c, p)$

- *self weigh* of the parent, which does not depend on the identity of the current target category *c*, but rather quantifies the importance of the candidate parent to be included in the target-domain taxonomy;
- *daughter-parent weight* that allows considering preferences related to the current target category c when choosing its parent.

For calculating both of these weights we suggest a *backward and forward looking* approach, aiming to look beyond the single local edge and consider a wider perspective of categories' ancestors and descendants when choosing the parent category at each step.

Self weight. When calculating the self-weight of a candidate parent p we consider the following criteria for its importance and relevance for the target-domain taxonomy. The category should:

- 1. Represent a concept or topic prominent within the target domain (local-relevance).
- 2. Represent a general, not too narrow topic or concept (local-importance).

²http://www.imdb.com/

- 3. Have a relevant category of high importance amongst its ancestors (look forward).
- 4. Have many important and relevant categories amongst its descendants (look backward).

We reflect the first two criteria in a local self weight (lsw) of a category, which we define as follows:

 $lsw(p) = freq_{domain}(p) + \frac{freq_W(p)}{depth(p)}$

where $freq_{domain}(p)$ is the domain frequency (as defined in 3.1) of category p, $freq_W(p)$ is the number of Wikipedia articles that belong to the category p or its subcategories in WCH, and depth(p) is the length of the shortest path from p to the root of WCH. This simple heuristic promotes categories frequent in the target-domain corpus, while being general enough to cover many Wikipedia articles and be placed not too far from the hierarchy root.

To address the 3^{rd} and 4^{th} criteria we define the

self weight of a candidate parent p as follows: $w_{self}(p) = lsw(p) + \frac{\sum_{a \in A} lsw(a)}{|A|} + \frac{\sum_{d \in D} lsw(d)}{|D|}$ where A is the set of p's ancestors and D is the set of p's descendants in WCH.

Daughter-parent weight. By introducing a daughter-parent weight we expect to improve the selection of the most appropriate path from WCH, which leads from the current category c to the root node. We do that by considering the preferences induced by a target category when choosing its parent. We note that different candidate parents of a target category tend to represent different contexts, and sometimes "senses", for the category. For example, Albert Einstein falls among others under the categories theoretical physicists, zionists and American vegetarians.

We suggest that a target category c can assign a score to its candidate parent p by means of directional distributional similarity (see Section 2.3) $sim(c \rightarrow p)$, calculated using a corpus of the target domain. This provides implicit context selection for c in the target domain and ensures that the most relevant parent is preferred. Similarly to self weight calculation, we suggest combining direct (local) scoring by the current target category for a candidate parent with transitive (backwardforward) scoring:

 $w_{daughter-parent}(c,p) = \frac{\sum_{b \in B} \sum_{f \in F} sim(b \to f)}{|B| \cdot |F|}$ where F is the "forward" set containing the can-

didate parent p and its ancestors in WCH and Bis the "backward" set containing the current target category c and all of its descendants in the current



Figure 3: A screen shot of a portion of a target-domain taxonomy automatically generated by our algorithm for the movies domain, based on IMDb keywords.

taxonomy tree T. We use descendants from T and not from WCH because deeper categories in WCH are proceeded before higher ones and thus at each step the current target category c represents not only itself, but also the target categories (if any) that have already selected c to be their parent and whose preferences when selecting their path to the root should also be considered.

3.3 Pruning

When a target subtree is extracted from WCH, we apply a pruning procedure in order to retain only the most relevant categories and obtain a taxonomy of a desired size. The size can be specified by the user as a parameter of the algorithm. We employ the following simple pruning procedure:

- 1. For each category calculate its sub-tree weight by summing its own and all its subcategories' domain frequencies.
- 2. Prune categories whose sub-tree weight is lower than a threshold. Define the threshold to be depth-dependent, requiring a higher sub-tree frequency for deeper levels of the tree.

Figure 3 shows a sample from a resulting taxonomy tree generated by our algorithm for the movie domain.

4 Taxonomy Editing Tool

Automatically-generated taxonomies are usually not accurate enough and thus human inspection and post-editing is practically a necessity. In this section we describe our taxonomy post-editing tool, which aims to help the editor to correct some of the decisions made by the taxonomy generation algorithm, while making her work efficient in terms of both time and the quality of the resulting taxonomy. We note that the intelligent assistance suggested by our support tool can be applied to improve the output of any taxonomy generation algorithm.

The utility of the tool can be demonstrated thought three typical editing scenarios: (1) pruning the taxonomy from irrelevant categories, (2) enriching important categories with additional subcategories, which were not included in the initial taxonomy and (3) moving categories placed under an inappropriate parent to another place in the taxonomy. Below we provide examples for these scenarios.

The tool allows generating first an initial taxonomy of a desired size and then supports standard browsing and editing operations over it, such as creating, deleting and renaming categories. For each category the tool displays its domain frequency and sub-tree weight (cumulative frequency, Section 3.3) as in Figure 3. These statistics help the editor in deciding whether to delete a category or perhaps to enrich it with additional subcategories if the current subcategories do not suffice. They also attract the editor's attention to problematic parts in the hierarchy. For example, the category *meat* received a high sub-tree weight (614), while counting only 58 occurrences in the targetdomain corpus. The editor will see that 500 out of 614 occurrences were contributed by the rabbit category, which should rather be a subcategory of animals in the video domain.

While it is relatively easy for the editor to notice that a category is placed under a wrong parent, identifying an appropriate parent category is more difficult. Similarly, it is not easy to identify which additional daughters should be added to a given category. The tool's on-demand assistance described below helps the editor in these situations by providing suggestions for alternative parent categories and suitable subcategories.

Figure 4 presents an example of the on-demand assistance offered to the editor after clicking on



Figure 4: Part of the assistance view for the *Children* category.

the *children* category. The category, along with its current path from the root, is displayed at the top, with assistance information below. We see that the *children* category was placed under the *nature* \rightarrow *time* \rightarrow *human development* category due to the biological "sense" of the word *children*, while in the video domain it would be more suitable to place this category under the *family* category.

The tool uses two sources for suggesting both parent categories and daughter categories - WCH and distributional similarity calculated over the target-domain corpus (Section 2.3). From Figure 4 we see that information from the two sources is complementary and each source has its pros and cons. Distributional similarity is more noisy, but allows the editor to better understand the characteristic contexts of the specified category in the target domain and adds relevant suggestions not found in WCH.

For the example in Figure 4 the editor will see that there were no alternative parent categories in WCH, which explains the system's failure in placing the *chidren* category. She might then move *chidren* under the *family* category, which is the first choice suggested by distributional similarity.

She might then want to check what interesting subcategories are suggested for *children*, which is an important category in the domain (over 1800 occurrences), but had no subcategories in the automatically generated taxonomy. She might decide to add the *orphan* category suggested by WCH, as well as *boys* and *girls* suggested by distributional similarity as subcategories of *children*. We note that the editor can add and move categories in a single click without leaving the assistance window.

4.1 Initial Evaluation of the Tool

We performed initial evaluation by performing the task of generating a small taxonomy of 100 categories for the video domain. Creating a taxonomy manually, given the initial set of keywords and their domain frequencies, took about 20 hours. Post-editing of the automatically generated taxonomy (by another person) by means of the editing tool was accomplished in about 5 hours. The editor requested an initial taxonomy about twice as large as the required one and edited it mostly by removing some of the categories. Dozens of categories were enriched with additional subcategories and some were moved under a different parent category using the tool's assistance (Figure 4). In addition, the taxonomy generated using the tool included interesting categories not present in the manually created one.

The tool documents all the editor's actions in a detailed log file to enable further analysis and evaluations, including quantifying human editing effort.

5 Conclusions and Future Work

In this paper we presented a novel algorithm and an implemented interactive system for automatic generation of target-domain taxonomies. The algorithm combines knowledge-based and corpusbased techniques by deriving a taxonomy from Wikipedia Category Hierarchy, while relying on corpus statistics and distributional similarity. The system includes a taxonomy editing tool, facilitating human post-editing by means of intelligent assistance.

Our initial evaluations showed considerable reduction of time needed to create a taxonomy using the tool comparing to manual taxonomy creation. In the future we plan to conduct elaborate user studies to evaluate the quality of the algorithm and the usefulness of the assistance provided by the tool.

Acknowledgments

This work was supported by the NEGEV project (www.negev-initiative.org). We would like to thank Sonya Liberman for her help with the WCH.

References

- G. de Melo and G. Weikum. 2010. Menta: inducing multilingual taxonomies from wikipedia. In *CIKM*.
- C. Fellbaum. 1998. *WordNet An Electronic Lexical Database*. MIT Press.
- M. Käki. 2005. Findex: search result categories help users when document ranking fails. In *CHI*.
- L. Kotlerman, I. Dagan, I. Szpektor, and M. Zhitomirsky-Geffet. 2010. Directional distributional similarity for lexical inference. *JNLE*, 16.
- R. Krishnapuram and K. Kummamuru. 2003. Automatic taxonomy generation: Issues and possibilities. In *IFSA*.
- S. P. Ponzetto and R. Navigli. 2009. Large-scale taxonomy mapping for restructuring and integrating wikipedia. In *IJCAI*.
- S. P. Ponzetto and M. Strube. 2007. Deriving a large-scale taxonomy from wikipedia. In *AAAI*.
- W. Pratt, M. A. Hearst, and L. M. Fagan. 1999. A knowledge-based approach to organizing retrieved documents. In *AAAI/IAAI*.
- E. Stoica, M. Hearst, and M. Richardson. 2007. Automating creation of hierarchical faceted metadata structures. In *NAACL/HLT*.
- F. M. Suchanek, G. Kasneci, and G. Weikum. 2007. Yago: A large ontology from wikipedia and wordnet.