Taxonomy Builder: a Data-driven and User-centric Tool for Streamlining Taxonomy Construction

John Hungerford[*], **Yee Seng Chan**[†], **Jessica MacBride**[†], **Benjamin M. Gyori**[‡], **Andrew Zupon**[◁], **Zheng Tang**[◁], **Egoitz Laparra**[◁], **Haoling Qiu**[†], **Bonan Min**[†], **Yan Zverev**[*], **Caitlin Hilverman**[⊕], **Max Thomas**[⊕], **Walt Andrews**[⊕], **Keith Alcock**[◁], **Zeyu Zhang**[◁], **Michael Reynolds**[*], **Mihai Surdeanu**[◁], **Steve Bethard**[◁], **Rebecca Sharp**[♠]

[*]Two Six Technologies, Arlington, VA, USA [†]Raytheon BBN Technologies, Cambridge, MA, USA
[‡]Harvard Medical School, Boston, MA, USA [◁]University of Arizona, Tucson, AZ, USA
[⊕]Qntfy, Arlington, VA, USA [♠]Lex Machina, Menlo Park, CA, USA
john.hungerford@twosixtech.com, msurdeanu@arizona.edu

Abstract

An existing domain taxonomy for normalizing content is often assumed when discussing approaches to information extraction, yet often in real-world scenarios there is none. When one does exist, as the information needs shift, it must be continually extended. This is a slow and tedious task, and one that does not scale well. Here we propose an interactive tool that allows a taxonomy to be built or extended *rapidly* and with a *human in the loop* to control precision. We apply insights from text summarization and information extraction to reduce the search space dramatically, then leverage modern pretrained language models to perform contextualized clustering of the remaining concepts to yield candidate nodes for the user to review. We show this allows a user to consider as many as 200 taxonomy concept candidates an hour to quickly build or extend a taxonomy to better fit information needs.

1 Introduction

Information extraction (IE), or the extraction of structured information from free text, is a sub-field of natural language processing (NLP) that is of keen interest to those outside of the NLP community. Practitioners who desire to mine information from text often set this up as a two-part task: (1) extracting the structured information from each document, and then (2) normalizing the extractions to be able to aggregate across a given corpus.

For this second step, often referred to as *grounding*, there are several approaches, but in industry and many domain-specific applications, the standard method for grounding is linking to a relevant ontology or taxonomy (Friedman et al., 2001; Srinivasan et al., 2002; Shen et al., 2014; Sevgili et al., 2020), i.e., a set of domain concepts and their hierarchical organization (and additional relations in the case of an ontology). However, the task of creating or maintaining these resources is laborious and never complete; as new documents are added or the use case shifts, there are concepts that are not well covered by the current taxonomy. As a result, typically a user must manually add new, relevant concepts to the taxonomy – a process which is both time-consuming and expensive.

Here we present a tool that is designed to streamline this process by using automatically gleaned text summarization analytics from the corpus itself, coupled with the power and expressivity of recent contextualized embeddings, to suggest candidate concepts to a human user during an interactive session. The user can accept, reject, or manipulate the suggested concept, then determine where it belongs in relation to other existing nodes. Taxonomies can be persisted for downstream use or a subsequent editing session. Our contributions are:

(1) We propose a data-driven approach to interactively build or augment a taxonomy with new concepts derived from the corpus of interest. The human user is at the center of our workflow and retains full control. Our approach first ranks and then clusters corpus concepts to provide the user with highly salient and coherent suggestions for new taxonomy nodes. In a web application, the user can act on the suggestions by adding, editing, deleting, or skipping them as desired. The tool is designed to be domain agnostic and interactive, with expensive steps performed in advance.

(2) We provide two case studies to demonstrate the utility of our approach, first, focused on a causal analysis of the drivers of food insecurity, and second, managing regional security (Section 5). We show that a user was able to process an average of 200 nodes per hour, 16% of which were added to the taxonomy. We find that the extended taxonomy results in an overall consistently higher grounding

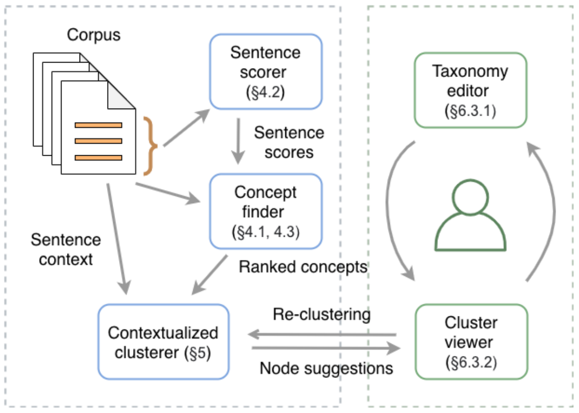


Figure 1: Overall architecture of our Taxonomy Builder, which combines offline pre-processing steps (the components in the left panel) with online user sessions (right panel).

confidence. We show that this increased confidence correlates with an increase in grounding correctness. Further, the updated taxonomy results in an increase in the number of causal relations between high-confidence grounded concepts.

2 Related work

There are several large, ontological resources, e.g., WordNet (Miller, 1995; Fellbaum, 2010), Cyc (Lenat, 1995), UMLS (Bodenreider, 2004), and SNOMED CT (Donnelly et al., 2006). However, resources such as these, even in aggregate, do not have coverage of all domains; instead, they need to be perpetually extended to cover new concepts (Powell et al., 2002; Ceusters, 2011).

To address the human cost of creating or maintaining taxonomies, many proposed approaches either rely on supervised data (Bordea et al., 2016; Mao et al., 2018; Espinosa-Anke et al., 2016, e.g.,) or perform unsupervised term extraction and clustering (Bisson et al., 2000; Drymonas et al., 2010, e.g.,). As supervised training cannot be assumed in real-world settings, our approach is more similar to the latter; we use unsupervised methods for concept discovery, ranking, and clustering. However, we keep the human in the loop to guide the process, critical for sensitive use cases such as military events, medical emergencies, etc.

Maedche and Staab (2001) similarly propose a tool that allows a user to guide a semi-automatic ontology creation process. However, our approach uses techniques from text summarization to filter candidate concepts and modern contextualized embeddings to more robustly handle multiple word senses and multi-word phrases to minimize the work that must be done by the user.

3 Architecture

Our approach for rapid data-driven taxonomy generation combines insights from several layers of text understanding to suggest highly relevant candidate concepts to a user, who decides how to use them (or not) to build the taxonomy they need. The architecture is shown in Figure 1.

Specifically, given a corpus of documents and optionally an existing taxonomy, we assign salience scores to each sentence based on keyword occurrence (Appendix A.1). From sentences with a sufficiently high salience, we extract multi-word expressions (noun and verb phrases) as potential phrases of interest. These are ranked with respect to each other using an extension of the TextRank algorithm (Mihalcea and Tarau, 2004) (Appendix A.3). The top-ranked concepts are encoded with contextualized word embeddings and clustered (Appendix B). Resulting clusters are presented to the user as suggested novel concepts; the phrases in the clusters can be thought of as *examples* of that concept. To ensure interactivity, these computationally expensive steps are done ahead of time, and the user need only load the pre-computed initial clusters.

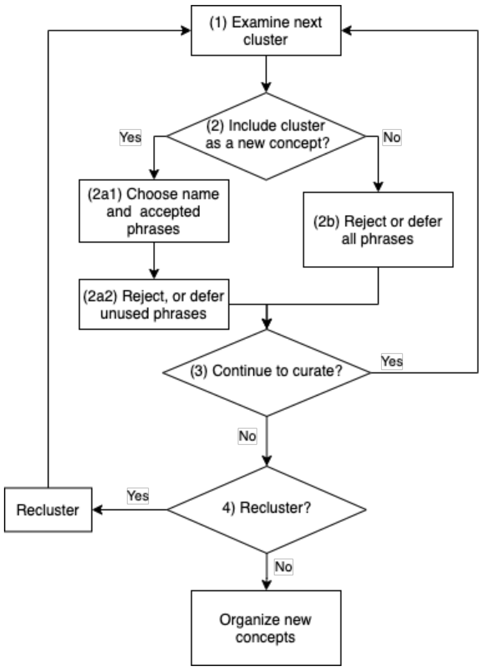
Once loaded, clusters are exposed to the user through our interactive web application. The user can then decide between a series of actions (Section 4); specifically, they can *accept*, *edit*, *skip*, or *discard* the node. Accepted nodes are then inserted into the current working taxonomy. The user continues to work through the system's suggestions until they are satisfied with the taxonomy.

4 Taxonomy builder workflow

Once phrases are ranked and clustered, they are passed to the user-facing interface, which implements a two-part workflow: (a) cluster curation and (b) concept organization.

4.1 Cluster curation workflow

We frame the **cluster curation** workflow as a potentially iterative process, where the user adds novel nodes to the taxonomy from the suggested clusters and then, if desired, submits unused phrases for reclustering and a subsequent iteration of curation. By eliminating phrases that are irrelevant or have already been associated with a taxonomy node, each iteration provides an improved basis for organizing the remaining phrases into new, potentially usable clusters. The workflow for a curation iteration is illustrated in Figure 2:



(1) Examine next cluster
(2) Include cluster as a new concept?
Yes
No
(2a1) Choose name and accepted phrases
(2b) Reject or defer all phrases
(2a2) Reject, or defer unused phrases
(3) Continue to curate?
Yes
No
Recluster
Yes
4) Recluster?
No
Organize new concepts

Figure 2: Cluster curation workflow.

1 User examines clusters in descending order of cluster score.

2 For each cluster, the user decides whether to include it as a taxonomy node, informed by its perceived relevance to the use case and its perceived added value, relative to existing nodes.

2a1 If the cluster is included, one phrase is selected for the node *name* and one or more phrases are chosen as concept *examples*; this can be done in bulk.

2a2 User can also *reject* or *defer* unused phrases from this cluster. Rejected phrases are excluded from reclustering, whereas deferred phrases are included if the user chooses to recluster.

2b If the cluster is not included in (2), the user can either reject or defer the entire cluster, or some of its phrases.

3 User decides whether to continue to curate or, if remaining clusters appear to be unlikely candidates for inclusion, to *defer* the remaining clusters.

4 Having completed an iteration of curation, the user either reclusters or moves on to the concept organization workflow.

4.2 Concept organization workflow

The **concept organization** workflow is: Examining each of the newly generated concepts in turn, the user 1) searches for an existing taxonomy node that would be a suitable parent to the new concept, 2) adds the concept to an existing branch if one is found or a new branch if none exists, and 3) updates any additional required concept metadata. Once all concepts have been organized within the taxonomy, the user publishes the new or augmented taxonomy.

4.3 User interface

The above workflows are carried out using a web application that includes a cluster viewer and a taxonomy editor. The **cluster viewer** displays the clustered concepts and allows the user to include or exclude clusters. The **taxonomy editor** allows the user to traverse the taxonomy tree, add or remove nodes, move existing nodes to different branches, and make changes to node metadata. The user can toggle between these at any time.

4.3.1 Cluster viewer

The cluster viewer consists of 1) reclustering controls, 2) a cluster list, and 3) a target node editor. The reclustering controls allow the user to submit new clustering jobs and access previously submitted jobs. The cluster list displays all clusters in descending order of score and contains controls for adding clusters to the taxonomy as concept nodes. The target node editor permits directly editing a new or existing taxonomy node corresponding to the given cluster (see Figure 3).

Each displayed cluster has a panel showing a) a node selector tool allowing a user to select an existing node or create a new node in the taxonomy as the "target" for the current cluster, b) a recommended concept name, initially populated by the first phrase within the cluster, c) a switch to either reject or curate the cluster (*curate* by default), d) the list of cluster phrases, e) phrases selected for inclusion, f) rejected phrases. Each cluster phrase has an option to *accept* it as an example, *reject* it, or *select it as the concept name*. The target node can be updated in the target node editor panel (3) and even moved in the taxonomy by updating its "parent" field. Through the inclusion of defaults, we ensure that the user does not need to treat all phrases, saving them time.

When the user first enters the cluster viewer, the application fetches and displays the initial clustering results. Once the user has included at least one cluster as a new concept by accepting a phrase, the option to recluster is enabled. When the user executes this option, the application submits a clustering job including all deferred phrases, clears the cluster results, and polls the clustering service for the job status. When the new clustering job is finished, the new cluster results are displayed and the job id is persisted in the application state for that user, allowing the user to follow the curation workflow over multiple iterations despite interruptions. Accepted clusters are automatically added to a top-level branch in the taxonomy named “clusters.” Any changes to a cluster’s name and accepted phrases automatically propagate to the

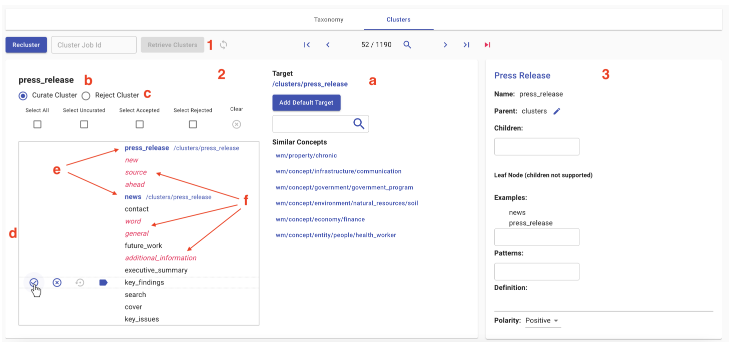


Figure 3: Cluster viewer user interface. Number and letter labels refer to the elements as described in Section 4.3.1.

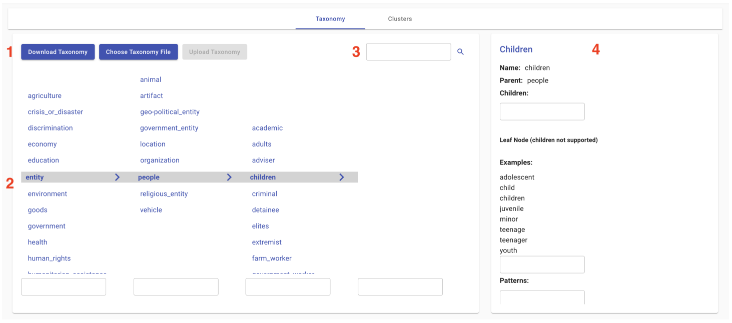


Figure 4: Taxonomy editor user interface. Number labels refer to the elements as described in Section 4.3.2.

corresponding concept's name and examples.

4.3.2 Taxonomy editor

The taxonomy editor consists of 1) import/export controls, 2) taxonomy explorer, 3) concept search widget, and 4) node editor (Figure 4). The import/export controls permit the user to upload a taxonomy and download the current state of a taxonomy as a taxonomy file. The taxonomy explorer provides a collapsed tree view of the taxonomy similar to multi-directory views common to many operating systems. This permits users to see a selected node, its siblings, its children, and its closest three ancestors and their siblings. Visible nodes can be clicked to become the new selection, allowing users to traverse the taxonomy in any "direction." This format was chosen because it displays a relatively broad cross-section of the taxonomy without sacrificing intelligibility and navigability. The concept search widget allows users to search for concept names and navigate to them without having to click through the explorer. The node editor allows users to update the concept name, update its parent, add and remove children, and update node metadata.

5 Usage Scenarios

We evaluate our tool through two use cases, both of which are motivated by DARPA's World Modelers program:[1] food insecurity and regional security. We detail both evaluations in Appendix C. The take-home messages from this evaluation were: (a) the users were able to process 200 candidates for taxonomy concepts per hour using the tool; (b) the updated taxonomies yielded increased grounding confidence scores, which indicate that the new taxonomies fit the data better, and (c) the new grounding confidence scores have increased correlation with grounding correctness.

6 Conclusions

We introduced a tool to streamline taxonomy construction and extension. Using techniques from text summarization alongside the benefits of modern pretrained language models, we automatically glean cohesive taxonomy node suggestions from the corpus itself. The user interface then allows users to quickly review suggestions and decide whether to accept or reject part or all of each. Through two case studies, we showed that this tool can be used to rapidly extend an existing taxonomy in a matter of hours, and further that the resulting taxonomy is a better fit to the domain of interest.

The software is included in the DART framework at: https://github.com/twosixlabs-dart/dart-ui.

[1] https://www.darpa.mil/program/world-modelers

References

Mehdi Allahyari, Seyedamin Pouriyeh, Mehdi Assefi, Saeid Safaei, Elizabeth D Trippe, Juan B Gutierrez, and Krys Kochut. 2017. Text summarization techniques: a brief survey. *arXiv preprint arXiv:1707.02268*.

Gilles Bisson, Claire Nédellec, and Dolores Canamero. 2000. Designing clustering methods for ontology building-the mo'k workbench. In *ECAI workshop on ontology learning*, volume 31. Citeseer.

Olivier Bodenreider. 2004. The unified medical language system (umls): integrating biomedical terminology. *Nucleic acids research*, 32(suppl_1):D267–D270.

Georgeta Bordea, Els Lefever, and Paul Buitelaar. 2016. SemEval-2016 task 13: Taxonomy extraction evaluation (TExEval-2). In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1081–1091, San Diego, California. Association for Computational Linguistics.

Werner Ceusters. 2011. Snomed ct revisions and coded data repositories: when to upgrade? In *AMIA Annual Symposium Proceedings*, volume 2011, page 197. American Medical Informatics Association.

Kevin Donnelly et al. 2006. Snomed-ct: The advanced terminology and coding system for ehealth. *Studies in health technology and informatics*, 121:279.

Euthymios Drymonas, Kalliopi Zervanou, and Euripides G. M. Petrakis. 2010. Unsupervised ontology acquisition from plain texts: The ontogain system. In *Natural Language Processing and Information Systems*, pages 277–287, Berlin, Heidelberg. Springer Berlin Heidelberg.

Luis Espinosa-Anke, Horacio Saggion, Francesco Ronzano, and Roberto Navigli. 2016. Extasem! extending, taxonomizing and semantifying domain terminologies. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Christiane Fellbaum. 2010. Wordnet. In *Theory and applications of ontology: computer applications*, pages 231–243. Springer.

Carol Friedman, Hongfang Liu, Lyudmila Shagina, Stephen Johnson, and George Hripcsak. 2001. Evaluating the umls as a source of lexical knowledge for medical language processing. In *Proceedings of the AMIA Symposium*, page 189. American Medical Informatics Association.

Benjamin M Gyori, John A Bachman, Kartik Subramanian, Jeremy L Muhlich, Lucian Galescu, and Peter K Sorger. 2017. From word models to executable models of signaling networks using automated assembly. *Molecular systems biology*, 13(11):954.

Douglas B Lenat. 1995. Cyc: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):33–38.

Hans Peter Luhn. 1958. The automatic creation of literature abstracts. *IBM Journal of research and development*, 2(2):159–165.

Alexander Maedche and Steffen Staab. 2001. Ontology learning for the semantic web. *IEEE Intelligent systems*, 16(2):72–79.

Debanjan Mahata, John Kuriakose, Rajiv Ratn Shah, and Roger Zimmermann. 2018. Key2Vec: Automatic ranked keyphrase extraction from scientific articles using phrase embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 634–639, New Orleans, Louisiana. Association for Computational Linguistics.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL)*.

Yuning Mao, Xiang Ren, Jiaming Shen, Xiaotao Gu, and Jiawei Han. 2018. End-to-end reinforcement learning for automatic taxonomy induction. *arXiv preprint arXiv:1805.04044*.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.

Patrick Pantel and Dekang Lin. 2002. Discovering word senses from text. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining (SIGKDD-02)*, pages 613–619.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Tammy Powell, Suresh Srinivasan, Stuart J Nelson, William T Hole, Laura Roth, and Vladimir Olenichev. 2002. Tracking meaning over time in the umls metathesaurus. In *Proceedings of the AMIA Symposium*, page 622. American Medical Informatics Association.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. 2008. *Introduction to information retrieval*, volume 39. Cambridge University Press Cambridge.

Ozge Sevgili, Artem Shelmanov, Mikhail Arkhipov, Alexander Panchenko, and Chris Biemann. 2020. Neural entity linking: A survey of models based on deep learning. *arXiv preprint arXiv:2006.00575*.

Rebecca Sharp, Adarsh Pyarelal, Benjamin Gyori, Keith Alcock, Egoitz Laparra, Marco A Valenzuela-Escárcega, Ajay Nagesh, Vikas Yadav, John Bachman, Zheng Tang, et al. 2019. Eidos, INDRA, & Delphi: From free text to executable causal models. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*.

Wei Shen, Jianyong Wang, and Jiawei Han. 2014. Entity linking with a knowledge base: Issues, techniques, and solutions. *IEEE Transactions on Knowledge and Data Engineering*, 27(2):443–460.

Suresh Srinivasan, Thomas C Rindflesch, William T Hole, Alan R Aronson, and James G Mork. 2002. Finding umls metathesaurus concepts in medline. In *Proceedings of the AMIA Symposium*, page 727. American Medical Informatics Association.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

A Concepts

For the process of selecting phrases from the corpus to serve as potential concepts of interest (or examples of those concepts), we use filters to both improve the speed of clustering and reduce the noise in the final clusters themselves.

A.1 Sentence salience

To prevent candidate phrases that are not salient to the core content of the documents, we use a simple *extractive text summarization* technique to identify the most salient or important sentences and assign each a score (Luhn, 1958; Allahyari et al., 2017).

We first lowercase document text and tokenize with `spaCy`,[2] ignoring stopwords and punctuation. We then extract keywords by calculating the frequency of all words, normalizing such that the most frequent has a value of 1. Finally, we calculate sentence salience scores by identifying the occurrence of keywords within a sentence, summing their frequency values, and normalizing these sums such that the most salient sentence in a document, i.e., the one with the highest sum, has a score of 1.

The sentence salience score quality was evaluated against a human rater. We selected random pairs of sentences, and for each asked our rater to determine which sentence was more important to the meaning of the document. A preliminary analysis with 10 news articles found that the human judgment agreed with the assigned scores 80% of the time. In cases of disagreement, typically the human chose a headline or summary sentence, whereas the algorithm chose a sentence with detailed but relevant information. From this preliminary analysis, we made two minor changes that made the model more robust to irregularities in documents: disregarding bullet points and subheadings. After making this change, rater judgments were nearly 100% aligned with the assigned salience scores.

A.2 Candidate phrases

To get candidate phrases from the sufficiently salient sentences, we process them with the CLU lab processors library.[3] We then select noun and verb chunks, splitting on coordinating conjunctions and trimming determiners from the edges. We note frequency and where they occur, keeping the 10k most frequent.

[2]`https://spacy.io`

[3]`https://github.com/clulab/processors` We use FastNLPProcessor, based on CoreNLP (Manning et al., 2014).

	Strategy 1	Strategy 2	Strategy 3
Expert 1	55%	30%	25%
Expert 2	70%	45%	25%

Table 1: Manual evaluation results (P@20) for three different strategies (Section A.3) for edge weights in the phrase graph. Strategy 1 uses $similarity$ alone, Strategy 2 uses $similarity \times PMI$, and Strategy 3 uses $similarity \times PMI \times frequency$. P@20 indicates what percentage of the concepts ranked in the top 20 were considered relevant for the use case by the corresponding expert.

A.3 Ranking candidates

These 10k phrases are ranked using an extension of TextRank (Mihalcea and Tarau, 2004), an algorithm inspired by PageRank (Page et al., 1999) that treats text as a graph and applies a graph-based ranking algorithm to surface keywords or phrases. The key extension in this effort is that nodes in the constructed graph are the phrases previously extracted, rather than full sentences, as in the original algorithm.

The algorithm consists of four steps: (1) Identify text units that best fit the task to be nodes in the graph; (2) Identify relations between units and draw the corresponding edges; (3) Iterate the graph-based ranking algorithm until convergence; (4) Sort nodes based on ranking scores.

As mentioned, in our implementation of TextRank, we consider our extracted chunks to be the text units (i.e., nodes in the graph) rather than complete sentences. We experimented with several options for defining edge weights, including word embedding similarity, Point-wise Mutual Information (PMI), and frequency of co-occurrence in the same sentence. (Mahata et al., 2018). For the embedding similarity, we represent each phrase as its GloVe embedding (Pennington et al., 2014), averaging embeddings of multi-word expressions, and compare with cosine similarity. Building on these three information sources, we compared three edge similarity strategies:

$$strategy_1 = cosine\_similarity(c_1, c_2) \quad (1)$$

$$strategy_2 = strategy_1 \times PMI(c_1, c_2) \quad (2)$$

$$strategy_3 = strategy_2 \times log(cooccur(c_1, c_2)) \quad (3)$$

where c_1 and c_2 are the phrases to be compared.

Using a small set of documents, we had two domain experts do a blind evaluation of the top-ranked phrases produced by the different strategies. As shown in Table 1, strategy 1 produces the best TextRank overall,[4] and so was chosen. To avoid a

[4]Post-hoc analysis showed that strategy 2 prefers infre-

fully connected graph, which slows down the TextRank algorithm dramatically, we set a threshold for the similarity scores[5] and for each phrase, we keep only edges for the 100 most similar neighbors.

After ranking the extracted phrases, the highest-ranked 5k are used for generating the node suggestion clusters (Section B).

B Clustering

After extraction, phrases are clustered into semantically cohesive groups to serve as taxonomy node suggestions. We use Huggingface transformers (Wolf et al., 2020) to obtain contextualized DistilBERT (Sanh et al., 2020) embeddings of each occurrence of each phrase. We then use Annoy[6] to perform time-efficient nearest neighbor search over these embeddings. For each phrase, we obtain a ranked list (in terms of cosine similarity) of the top-k most similar phrases as input to the clustering.

To cluster the phrases, we employ an algorithm based on the CBC algorithm (Clustering By Committee) (Pantel and Lin, 2002), which uses average link agglomerative clustering (Schütze et al., 2008, Ch. 17) to recursively form cohesive clusters that are dissimilar to one another. For each cluster c that is formed, the algorithm assigns a score: $|c| \times \text{avgsim}(c)$, where $|c|$ is the number of members of c and avgsim(c) is the average pairwise cosine similarity between members. This score reflects a preference for larger and cohesive clusters. We then rank the clusters in decreasing order of their cluster scores, prioritizing the most effective and cohesive clusters to the user for selection and addition to the taxonomy.

C Usage Scenarios and Discussion

We evaluate our tool through two use cases, both of which are motivated by DARPA's World Modelers program.[7] The first use case focuses on food insecurity, which is a complex domain, spanning several disciplines including economics, government policy, agriculture, etc. The second use case addresses regional security, an equally complex domain.

C.1 Use case 1: food insecurity

For this use case, the user[8] was provided with an initial taxonomy[9] created for the DARPA World Modelers program, and used the tool to perform cluster curation and taxonomy editing for 2 hours. In this time, the user was able to curate 400 clusters. Of those 400, 65 were chosen for inclusion as taxonomy nodes, 18 were *deferred* for reclustering, and the remaining 317 were *rejected* entirely. After this curation, the resulting taxonomy was compared against the original.

For this case study, we use a corpus of 472 documents from the food insecurity domain (government and NGO reports, news articles, etc.). We perform IE using the Eidos causal information extraction system (Sharp et al., 2019), resulting in 209,352 extracted concepts related to food security.

We then attempt to ground each concept mention to the taxonomy, assigning a grounding confidence score. There are many ways of assigning a confidence score. Here, we create a vector representation for the mention to be grounded and of each node in the taxonomy by averaging the GloVe embeddings for each non-stop word in the mention's text span and the taxonomy node's examples, respectively. We then assign each mention to the node that is closest in terms of cosine similarity.

We analyze extracted mentions and their grounding before and after the taxonomy update. Of the $209,352$ concepts extracted, $170,488$ were grounded before the update and $170,539$ were grounded after.[10] Further, after the update, $48,404$ concepts (28% of grounded concepts), were grounded to a different taxonomy term. For example, the phrase *ethnic-religious divisions* was originally grounded to the taxonomy concept `crisis_or_disaster/conflict/hostility` (with a score of 0.48), but after the update, it is grounded to `ethnic_conflicts` (with a higher score of 0.56).

Of the groundings that changed, for $47,518$ the confidence increased after the taxonomy update (98% of those that changed). Figure 5 shows the distribution of the changes in grounding scores attributed to the taxonomy update. Importantly, we observed that this increase in grounding confi-

quent phrases that aren't descriptive of the overall topic, e.g., *intergovernmental panel* and *environ*, whereas strategy 3 oppositely adds frequent phrases, e.g., names of countries, which the experts deemed not useful for taxonomy construction.

[5]We found 0.0 to be both a useful and intuitive threshold.

[6]https://github.com/spotify/annoy

[7]https://www.darpa.mil/program/world-modelers

[8]One of the authors served as the tool's user.

[9]https://github.com/WorldModelers/Ontologies

[10]The Eidos system has an internal filter that doesn't produce groundings when the confidence is below 0.2.

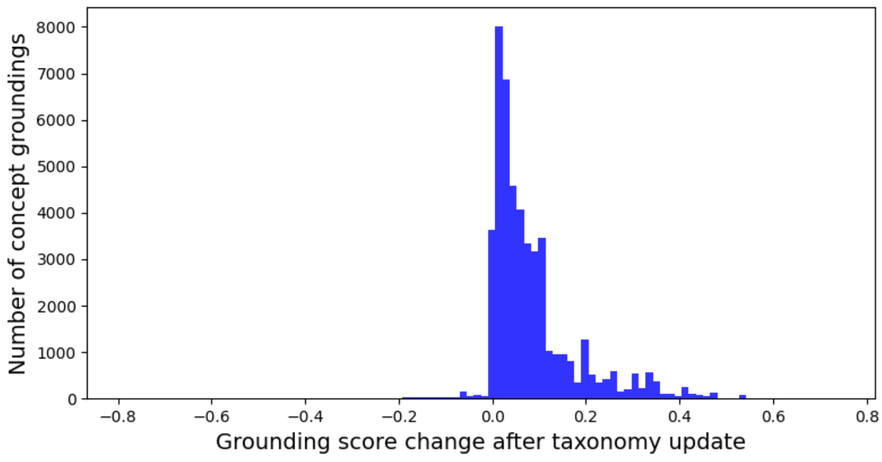


Figure 5: Histogram of changes in grounding scores associated with specific concepts after the taxonomy update in use case 1.

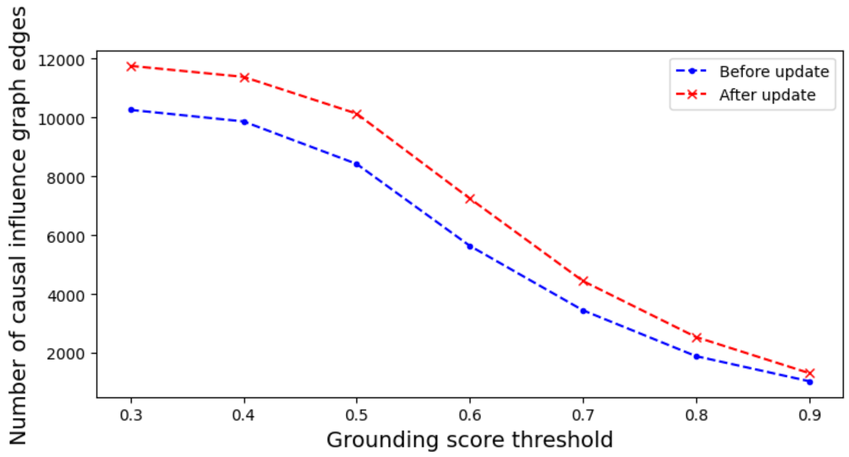


Figure 6: The number of causal influence graph edges over concepts obtained before and after the taxonomy update at a given grounding score threshold in use case 1.

dence correlates with grounding correctness. To verify this, we measured the Pearson correlation between grounding confidence and correctness (which is represented as a Boolean variable, i.e., correct/incorrect grounding) for 42 randomly selected concepts. The Pearson correlation values were 55.15% for the original taxonomy vs. 59.37% for the updated one, a relative increase of 7.6%.

Next, we use INDRA (Gyori et al., 2017; Sharp et al., 2019), an automated model assembly system, to assemble each set of causal influence relations (before and after taxonomy updating) into a causal influence graph by aggregating relations with matching groundings. We find that the number of edges in the causal influence graph increased from $11,072$ to $12,720$ after the taxonomy update, and further, we show in Figure 6 the number of edges in each influence graph between nodes whose grounding scores are above a given threshold. As shown, after the taxonomy update, the influence graph is consistently larger and covers more taxonomy terms compared to before, with higher confidence.

C.2 Use case 2: regional security

This use case was performed by a group of expert analysts outside the tool's developer team as part of a DARPA program evaluation. The analysis attempted to model a complex regional crisis scenario, using IE from documents to construct models of causal influences of security concerns surrounding Kenya's 2022 elections.

Users started from an initial taxonomy[11]. 10k documents relevant for the use case were identified by the organizers of the evaluation to seed the taxonomy extension process. Users were then given access to the tool to perform cluster curation and

[11] https://github.com/WorldModelers/Ontologies

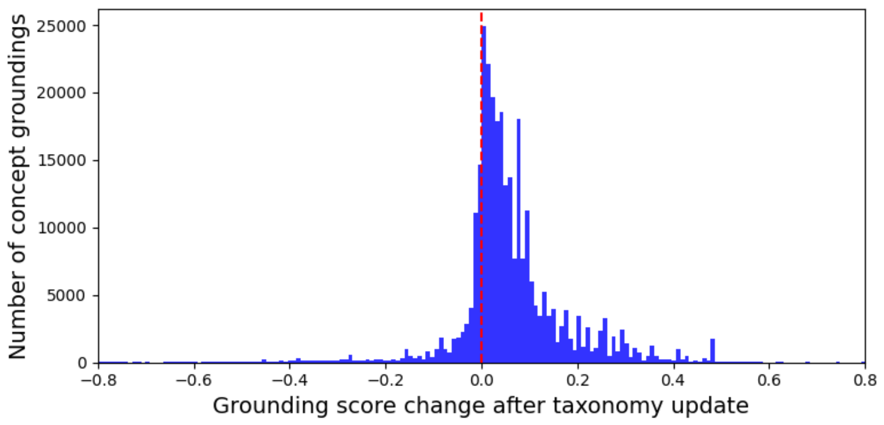


Figure 7: Histogram of changes in grounding scores associated with specific concepts after the taxonomy update in use case 2 (the red line provides vertical axis at 0 score change for clarity).

taxonomy editing over the course of multiple days.

Similar to use case 1, to evaluate the effect of taxonomy changes, we performed IE using Eidos on the seed corpus. This resulted in a total of $1,205,628$ concepts extracted from the 10k document corpus. We then grounded each extracted concept – using the approach described for use case 1 – with respect to the taxonomy both before and after the update. We found that $297,405$ of the extracted concepts (24.6% of all extracted) were grounded to different taxonomy entries after the update. Of these, $247,113$ (83%) were grounded with a higher score compared to before (see Figure 7 for the distribution of score changes).

Using INDRA, we then assembled causal relations that were extracted between concepts from the corpus into a causal influence graph before and after the taxonomy update. In both cases, we applied a grounding score threshold of 0.6 to retain concepts grounded to a taxonomy term with high-confidence. We found that the number of nodes in the graph increased from 337 to 451 (an increase of 33.8%) and the number of edges grew from $23,274$ to $29,562$ (a 27.6% increase) after the update. Overall, we again found that the taxonomy update resulted in a larger causal influence graph at a given level of confidence.