Bootstrapping Ontology Graphs

Anonymous authors

Abstract

Ontology graphs are graphs in which the nodes are generic classes and edges have labels that specify the relationships between the classes. In this paper, we address the question: to what extent can automated extraction and crowd sourcing techniques be combined to bootstrap the creation of comprehensive and accurate ontology graphs? By adapting the state-of-the-art language model BERT to this task, and by leveraging a novel crowd sourced relationship selection task, we show that we can use this approach to bootstrap the ontology graph creation for further refinement and improvement through human effort.

1. Introduction

Ontology learning from text is the problem of extracting classes, relationships, and their complete logical definitions [6]. An ontology graph is a subset of an ontology in that it focuses primarily on representing classes and class to class relationships. For example, given the sentence: "*Cytoplasm consists of jelly-like cytosol within a cell*", its ontology graph contains nodes for *cytoplasm, cytosol* and *cell*, with an edge between *cytoplasm* and *cytosol* labeled as *material*, and an edge between *cell* and *cytoplasm* labelled as *is inside*. In addition, it contains a node *jelly-like* connected to *cytosol* using the label *viscosity*. The relationships such as *material, is inside and viscosity* are not directly mentioned, and there could be many different ways to express these relationships in text.

Ontology graphs are different from fact graphs even though the boundary between the two is not sharp. For example a fact graph may have nodes representing specific real world entities such as *Joe Biden, United States*, and the edges such as *president of*. Automatically extracting ontology graphs from text is considerably more complex than extracting fact graphs from text for at least, three different reasons. First, it is unclear how to configure a self-supervised ontology learning pipeline to give high performance on this task. Second, there is limited naturally occurring training data available about relationships between classes that makes it difficult to even build a supervised learning system. Finally, as we see in the above example, quite often the classes and relationships that are desirable for extraction are not directly mentioned in natural language.

Comprehensive and precise ontology graphs are important for applications in education [3], law [18], and finance [1]. As a concrete example, ontology graphs when used in conjunction with an intelligent textbook lead to significant educational improvements [3]. (In the Appendix A, we include an illustration of an existing intelligent textbook.) The requirements of this application are very different from related problems such as answering multiple choice exam questions [7]. For example, an intelligent textbook glossary page for a class shows its super classes and sub classes. In a multiple choice exam, there are likely to be only a few questions about this relationship, and most certainly, correctly knowing all such relationships is not required. In contrast, when a student reads the intelligent textbook glossary, the subclass and super class relationships need to be all present and correct, and hence the need for an accurate and comprehensive ontology graph.

In this paper, we address the question: how can we bootstrap the creation of accurate and comprehensive ontology graphs? We answer this question by reporting the baseline performance of the current deep learning language model BERT for entity and relation extraction, and by defining a data labeling task to generate the training data necessary for further improving their performance. An innovative aspect of our data labeling approach is that it is designed to be educationally useful to students studying from a textbook. By deploying the labeling task on a textbook reading platform it will be possible to gather huge amount of training data at little to no cost. Once fully trained, this new approach has the potential to become self-supervised in the long term for creating precise and high coverage ontology graphs. Resulting automation will enable the creation of large libraries of intelligent textbooks and development of applications in related disciplines such as finance and law.

We will begin by describing our approach, and then present its experimental evaluation. We will conclude with a summary and a discussion of related work.

2. Overview of the Approach

Creating an ontology graph involves identifying classes and the relationships between them. Towards this goal, we leverage the well known tasks of entity and relation extraction as shown in Figure 2 interspersed with human validation. We will explain the background and rationale for our approach next.

A concrete target for the required ontology graph is a knowledge base, called KB Bio 101 [4], which was manually curated by domain experts and was found extremely useful in an intelligent textbook [3]. Class definitions in KB 101 used a rule language [5]. As automatically extracting rule definitions is outside the reach of automated methods, we have focused our initial work on identifying only direct class level relationships.

Automated Term Extraction (ATE) is a heavily researched task [14] which is a natural candidate for identifying the classes that should be captured in an ontology graph. Our initial experiments showed that off-the-shelf tools such as spaCy [13] had a precision and recall in the range of 0.20 on this task, and hence, an approach that can learn the peculiarities of different domains was required. As textbooks come with a hand-curated glossary, the terms in the glossary can provide the training data necessary for automatic extractors. ATE is different from the closely related Named entity recognition (NER) task [14] which

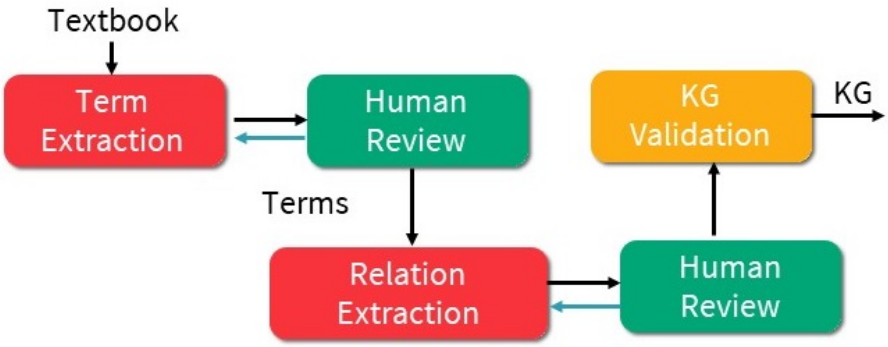

Figure 1: Ontology Graph Construction Pipeline

extracts concrete entities and labels them with a small number of classes such as person, location, company etc.

The choice of relationships to be represented in our ontology graph is based on the empirical evidence from KB Bio 101 and by recent work on linguistic analysis [10]. We retained 20 of over 100 relationships used in KB Bio 101 that were used most often, and dropped the ones that were found confusing (e.g., *object*, *base*, etc.) or infrequently used. The relationships we currently support include taxonomic relationships for classes and instances; structural relationships such as *has part*, *has region* and *material*, spatial relationships such as *is inside* and *is above*, functional relationships such as *has function* and *facilitates*, participant relationships such as *raw material* and *result*, event structure relationships, such as *subevent* and *next event* and causal relationships such as *enables* and *prevents*. We also allow the possibility that no direct relationship may exist between two terms.

Automatic relation extraction (ARE) has been extensively used for extracting relationships from text [14]. We used the data from KB Bio 101 and weak labeling methods for producing training data. In the experiments that we report here, the current ARE techniques do not have a high accuracy at this task. The primary bottleneck in improving the ARE methods is lack of training data. Traditional approaches to training data development have relied on custom labeling tasks [2]. As our goal is to minimize the costs, we designed a relationship selection task (RST) in which a student studying from a textbook engages in an educationally useful concept mapping activity [11] and chooses the correct relationship between two classes appearing in the textbook. We will deploy RST on a large open-source textbook reading platform that attracts several million students every year.

The relation extraction step. shown in figure 2, encapsulates both RST and ARE. In the early stages, we envision the RST to play a bigger role in creating the ontology graph, but as additional data accumulates, ARE would become more predominant. The human review step for both term and relation extraction refers to a review by an expert to ensure a highly accurate ontology graph. The backward arrows indicate that as human experts perform validation, it can be fed as training data to further improve the automated extraction. Future version of the system will include an active learning component [25] which would seek human input only for those cases where the confidence in the automated extraction is

low. Knowledge validation refers to the step of verifying the ontology graph that connects different class level relationships into a global whole. This step is outside the scope of the present paper.

For the rest of the paper, we will focus on describing the ATE, ARE and RST modules. In the current system, the relationship extraction is performed primarily through RST. Our ARE results establish a baseline performance which we hope to improve in our future work by leveraging the data collected through RST.

3. Tool Development

In this section, we will describe the design and implementation of the ATE, ARE, and the RST components of the ontology graph construction pipeline.

3.1 Automated Term Extraction

We formulate the term extraction problem as a supervised learning task where input is a textbook sentence, and the output is a label for each word based on the BIOES labeling scheme [21] indicating which of the word sequences constitute a term.

- Input: `['All', 'cells', 'have', 'a', 'cell', 'membrane', '.']`
- Output: `[ 'O', 'S', 'O', 'O', 'B', 'E', 'O']`

In the above sentence 'All' is labeled by 'O' which denotes other, 'cells' is labeled by 'S' which denotes a single word term, 'cell' is labeled by 'B' indicating the beginning of a term, and 'membrane' is labeled by 'E', indicating the ending of a term. ('I' is used to label a word that is in a term, but neither its beginning or the ending word.)

We fine tuned the cased base BERT model [8] for the term extraction task. Our fine-tuning involved first training the model on the textbook corpus, and then adding a fully connected softmax output layer that produces a probability distribution across the five BIOES tags for each token. The predicted tag for that token is then just the tag with the highest probability and the tag probabilities can be used to form confidence scores for the different predictions. We undertook our implementation using Pytorch [20], huggingface transformers bert_base_cased implementation of BERT [29], used commodity hardware, and standard techniques for optimizing the model.

3.2 Automated Relation Extraction

We focused on three of the most important relationships which also had the most training data available: taxonomic, structural and synonyms. A taxonomic relationship captures the *subclass* or *superclass* relationship between two terms. A structural relationship captures the *has part* or *has region* relationship between two terms that are entities. The *has part* relation is used for relating to physical entities, and *has region* relation is used to relate two spatial entities.

We formulate the relation extraction problem as a supervised learning task where the input is a sentence with two terms identified in the sentence with tags such as TERM1-START, TERM1-END, etc., and the output is one of the six relations (subclass, superclass,

has part/has region, has part of/has region of, synonym, OTHER). Here is an example of such input-output pair.

- Input: `['All', '[TERM1-START]', 'cells', '[TERM1-END]', 'have', 'a', '[TERM2-START]', 'cell' 'membrane', '[TERM2-END]', '.']`
- Output: `has part / has region`

We use a new variant of BERT known as BERT-EM [26], which modifies a pre-trained BERT model for supervised relation classification. In BERT-EM four new additional tokens are added to denote the start and end of both terms from the term pair in the sentence. These new tokens are shared across every sentence and thus allow a generalized representation of the locations of the two terms in a term pair to be learned across sentences. This modified input sequence is then run through a pre-trained BERT base encoder. The final encoder layer representations for the TERM1-START and TERM2-START tokens are then concatenated and fed into a linear layer that is run through a softmax to produce a probability distribution over relations. We implemented all models and training code using Pytorch and Python. We used the huggingface transformers bert_base_cased implementation of BERT [29]. We used commodity hardware and standard optimization techniques.

3.3 Relationship Selection Tool

We developed a relationship selection tool to guide the user in choosing the correct relationship between a pair of terms in the context of a sentence. The RST supports all twenty of the relationships that were mentioned in Section 2. The intended user of RST is a college student studying a particular discipline. The user is first asked to read a section from the textbook and then to undergo a short training on the relationships. We designed the training using simple common-sense examples that new users would find easy to understand. For example, we explain the *is inside* relationship using a visual in which a cat is shown hiding inside a box. (A detailed walk through of the training is available in Appendix C. The final version of the paper will be accompanied by an online demonstration.) As a concrete example, consider an example where the user is asked to relate the terms "cytoplasm" and "nucleus". The user first chooses the appropriate relationship family for the terms, including taxonomic, spatial, and component-based relationships. In this example, the correct relationship family is a spatial and choosing this option takes them to a second set of options to specify which spatial relationship is correct. They have an option to flip the order of the terms to ensure that the chosen relationship applies in the correct direction.

4. Experimental Setup

In this section, we consider the data that we used to test the ATE, ARE, and RST modules.

4.1 Data for Automated Term Extraction

We used 10 open source textbooks (as listed in Table 4 in the Appendix B) across multiple science subjects such as Physics, Chemistry, Biology, Astronomy and Anatomy. In addition, we used a textbook published by Macmillan Learning entitled LIFE biology [24]. Each of

these textbooks contains a glossary. We used the entries in the glossaries as training data for the terms that need to be extracted.

We then used the glossary terms from each textbook to automatically insert the BIOES tags into the sentences of that book. We handled lexical modifications such as pluralization, capitalization, etc., by using spaCy [13] to pre-process the sentences and terms so that we could match the terms on their lemmatized forms. We also handled a few special cases such as acronyms and hyphenation.

We used the data from eight textbooks except the OpenStax Biology 2e (Bio2e) and LIFE in the training set. To evaluate our models, a Biologist hand labeled a development set consisting of three sections from Bio2e (Chapter 4 section 2, Chapter 10 sections 2 and 4) and a test set consisting of Chapter 39 from LIFE. Finally, we removed all sentences from the training set that contained any of the terms from the development and the test sets. The sentence and term counts for the resulting splits are shown in Table 4.

4.2 Data for Automated Relation Extraction

To train a model for the relation extraction task, we need training data that has sentences with term pairs tagged, and the correct relationship identified between those terms. We created a data set where sentences were extracted from two Biology textbooks: Bio2e and LIFE. We used a term list obtained by combining the glossary of Bio2e and all of the biology terms from KB Bio 101 [4]. We used spaCy to tag each sentence using these terms. Tagging was done using lemmatized forms to account for lexical variations.

To associate a relationship between the pair of terms appearing in each sentence, we employed weak supervision using Snorkel [22]. This process involves three steps. First, we defined a set of labeling functions where each labeling function takes as input a sentence-term pair, and outputs a relation label or a special ABSTAIN response indicating that the labelling function cannot make a judgement. Second, we apply these functions to each term pair appearing in the input sentences producing a set of labels. Finally, we aggregated the predictions to obtain a single label. The functionality needed for step 2 is provided in Snorkel. We describe the first and the third steps in greater detail.

We used three categories of labeling functions: pattern-based, term-based and distance supervision-based. The pattern-based functions are similar to the Hearst patterns [12] with one difference — instead of operating on the input sentence, they operate on the dependency parse of the sentence produced by spaCy. For example, a *has* pattern label function labels two terms in a sentence with the has part/ has region relationship if the path between them in the dependency parse consists of a single step which is labeled as “has” or “have”. The term-based functions operate purely on the syntactic form of the two terms and ignore the sentence they appear in. For example, if two terms end with the same base word but one has an additional modifier in front of it, this suggests a taxonomic relation (e.g., “eukaryotic cell”*subclass*“cell”). The distance supervision function uses the existing hand curated KB Bio 101 by looking up the relationship between the two terms in the KB.

Snorkel provides two different ways of aggregating the labels produced by the individual functions: hard labels and soft labels. We can produce a single hard label for each data instance by taking a majority vote across labelling functions. This means that the relation that was labelled by the most labelling functions becomes the label for that instance. If

there is a tie or no label functions applied, then the label is ABSTAIN. For producing a soft label, we can produce a probability distribution across relations by using the label model included with Snorkel [23]. The label model learns a latent underlying confidence for each label function using the co-occurrences amongst each of the label functions produced in step 2. Label functions that are estimated to be more reliable are given more weight and those that are less reliable are given less weight. Thus when multiple label functions label the same instance, their label votes are combined based on their reliability estimates. For example, a term pair that receives a certain label from three label functions will have a higher confidence in that label than the one that receives a single label or conflicting labels. We experimented with both soft labels and hard labels.

We split all available sentences into a training, development, and test set. The development set consists of all sentences from the cell structure and cell cycle chapters of both textbooks (Ch. 4 and 10 of Bio2e, Ch. 5 and 11 of LIFE). The test set consists of all sentences from Ch. 39 of LIFE. All term pairs in both the development and test set were removed from the remainder of the data set which then formed the training set.

4.3 Data for Relationship Selection Task

As the ARE task covered only a small number of relationships, we tested RST in conjunction with the output of the ATE. We selected Sections 4.2 and 10.2 from Bio2e, and Section 14.1 from the OpenStax Psychology 2e textbook. We used ATE to identify all the terms in each section. As the precision and recall of the ATE is not perfect, two independent biologists on the team validated the terms. We then parsed the chosen section into individual sentences and automatically identified all term pairs that existed in each sentence. A sentence that contained N terms would have $\binom{N}{2}$ possible pairings, with each pairing considered to be a single task. Using the RST, the tasks were presented to users on a crowd sourcing platform.

Workers have varying degrees of competency and tasks have varying degrees of difficulty. Therefore, taking a simple majority vote to determine the correct task label can lead to significant estimation errors. To overcome these issues, we leveraged a sophisticated denoising algorithm [27, 9] which takes the worker ability and task difficulty into account to determine the denoised relation label identified by each worker.

5. Results and Analysis

We now describe the results and analysis for each of the ATE, ARE and RST modules.

5.1 Automated Term Extraction

Term extraction is a highly imbalanced classification task (the vast majority of words are not terms). For the test set, the model had a recall of 0.60, a precision value of 0f 0.51 and an F1 score of 0.55. Most interesting outcome of the experiment is the failure analysis presented below. From the failure sources, by adequately addressing lexical knowledge and general terms, we believe this performance can be significantly boosted. Handling terms that are not externalized in natural language is an inherent limitation of automatic methods, and will continue to require human intervention.

Lexical Knowledge: Lexical knowledge such as pluralizations, acronyms, different verb tenses, nominalizations, etc., are essential for good performance on ATE. Even though we were able to leverage some lexical knowledge in spaCy, our analysis of the failing cases revealed that there are numerous pieces of domain-specific lexical knowledge: chromatin vs chromatid, centromere vs centromeric. A textbook-specific linguistic resource, similar to Wordnet [19], is essential for better performance on ATE.

General Terms: Some terms in the ontology graph need to be defined in the textbook glossary, and hence, will not be present in the training data. For example, consider the term "microscope". For a first-year college level textbook, we can expect the student to know the meaning of a microscope, and we need not define it in the glossary. But in our ontology graph, we may have an experimental process in which microscope is the primary instrument used, and we would like it to be a node to specify that experiment. Therefore, *microscope* needs to be considered as a key term. The general terms such as microscope need to be curated for every textbook. Alternatively, we can envision a scenario in which every textbook is also accompanied by a glossary of prerequisite terms that it assumes a student to know. If such lists of prerequisite terms were available, they can be used to enhance the training data.

Terms not directly mentioned in text: There are several categories of terms that are not directly expressed in the natural language, and we consider two examples. The text often describes concepts without actually assigning them a name, for example, *mechanisms that ventilate the environmental side of those surfaces with air or water*, which can be simply referred to as a *ventilation mechanism*. Identifying such key terms poses a dual challenge. On the one hand, the term mentioned in the text is a long phrase which the automated extractor fails to pick up, and on the other hand, even if the extractor picks out the phrase, there is a challenge of associating a succinct name with the key term. There are many examples where the textbook describes different aspects of the same term which warrant a separate representation in the ontology graph, but it refers to them using the same term. For example, the psychology textbook first defines stress as a stimulus response mechanism, and then it defines stress as a physiological response. In both cases, it uses the word *stress* to define these two separate models which require distinct names in the ontology graph. Explicating and naming such indirectly stated concepts will continue to require some human intervention.

5.2 Automated Relation Extraction

We calculate precision, recall and F1 scores at the level of term pairs, that is, as long as the model predicted one of the six desired relationships between a term pair, we included the prediction in our calculations. We compared the model's prediction with the hand-labeled predictions to calculate the desired metrics. The results are shown in Table 1. If we put greater emphasis on recall than on precision, we can see that the model with soft labels performs better than the model with hard labels.

The results for individual relation specific classes are shown in Table 2. There are three noteworthy takeaways from these results. The results for recall are much higher than the results for precision. As previously noted, this is desirable because it reduces the effort in looking for relationship even though we may have to invest some effort in correcting

Table 1: Relation Extraction Model Comparison.

Source	Precision	Recall	F1
BERT-EM Hard Labels	0.43	0.66	0.52
BERT-EM Soft Labels	0.30	0.76	0.43

Table 2: Relation Extraction Soft Label Results.

Relation	Precision	Recall	F1
subclass	0.32	0.88	0.47
superclass	0.53	0.76	0.62
synonym	0.25	1.00	0.42
has part/has region	0.15	0.50	0.22
has part of/has region of	0.17	0.80	0.28

the wrong results. Second, the metrics for the taxonomic relationships are higher than the corresponding metrics for the structural relationships. This is primarily because the training data for taxonomic relationships is much better than for the structural relationships. As the training data improves, we anticipate the relation extraction performance for those relations to improve. Finally, the precision and recall values need to be at least above 0.5 before we can directly leverage this approach to relation extraction. We next discuss how we can further improve the relation extraction performance.

Expanding labeling sources: The primary bottleneck in improving the performance of relation extraction is the limitation of training data. There are four possible avenues for cost effectively expanding the training data. First, we can add more weak labeling functions using more patterns or distance supervision sources. Second, we could leverage Wordnet which has many concepts found in textbooks and can increase coverage of the training data. Third, Wikipedia and Wikidata [28] have a lot of information on academic subjects that could be leveraged to expand the training data. Finally, we could leverage crowd sourcing to inexpensively obtain relation labels. The input provided by humans is clearly complementary to weak labeling functions. Crowd sourcing can be used in a targeted manner to fill in the gaps where programmatic labels are lacking.

Graph post processing and inference: The model produces a list of triples, but it is unclear exactly how comprehensive/coherent this set is until they are all connected into a global graph. In doing so, graph processing techniques could potentially be employed to prune false positive relations and to infer false negative relations. In this way, the relation extractor from text could act as a seed graph that could then be modified to greater accuracy.

5.3 Relationship Selection Task

There were a total of 192 tasks for Bio2e Section 4.2, 229 tasks for Bio2e Section 10.2, and 64 tasks for Section 14.1 of Psych 2e textbooks labeled by the crowd workers. From the labeled tasks, we randomly selected 50 tasks with their labels from the denoised output for each section and presented these to two subject matter experts to verify if the labels that

Table 3: Validation of Relationship Selection

Textbook section	Precision when either SME agreed	Precision when both SMEs agreed
Bio 4.2	0.98	0.70
Bio 10.2	0.96	0.81
Psych 14.1	0.98	0.67

we obtained through our experiment were, in fact, correct. We show the results in Table 3. The precision for the case when at least one of the SMEs agreed with the crowd workers was exceedingly high, i.e., 0.96 or higher. The precision if we require both SMEs to agree with the crowd workers was much lower, i.e/, 0.67 or higher. We believe that with future work on better training and guidance for the expert SMEs, and refinements to the RST tool, a precision of 0.9 or higher is achievable.

6. Conclusions and Future Work

Our work advances the most closely related prior work on creating ontology graphs from textbooks [17, 2] in three important ways. First, our labeling task is designed in a way that it can be incorporated into the textbook reading by students making it much more practical across large libraries of textbooks. Second, we handle a much larger set of relationships that include taxonomic, structural, functional and causal relationships. Finally, we are leveraging the recent success of self-supervised deep learning and pre-trained language model so that similar underlying infrastructure could be used for both entity and relation extraction.

As our goal is to produce accurate and comprehensive ontology graphs, both automated and crowds sourcing methods have inherent limitation unless they are sufficiently moderated by human expert validation. Therefore, we view the methods presented here as bootstrapping that can give us a head start as we start to develop a new ontology graph. As we gather more validated data with the help of crowd workers, we anticipate that ARE will improve to a point where it becomes a self-supervised learning system in the long run. In the mid-term, we anticipate that we will be able to use active learning so that we rely on crowd input only when the prediction confidence is very low.

We note that the approach outlined here only gives us class level relationships between nodes in an ontology. An ontology contains much deeper relationships between instances. For example, a chromosome is inside a cell only when it is a part of that cell. Chromosomes can be removed from a cell, and in that case, the *is inside* relationship does not hold true between a chromosome and a cell. An ontology is able to capture such nuances through a proper use of quantification in rule definitions. Automatically capturing rules is an avenue for future work.

Finally, there are many applications in law and finance that are similar to textbooks that require comprehensive and precise ontology graphs. We therefore, argue, that innovative combinations of automation and human methods in which the precision and coverage are held to a high bar is wide open area for exploration for automated knowledge base construction.

References

[1] Mike Bennett. The financial industry business ontology: Best practice for big data. *Journal of Banking Regulation*, 14(3):255–268, 2013.

[2] Jonathan Berant, Vivek Srikumar, Pei-Chun Chen, Abby Vander Linden, Brittany Harding, Brad Huang, Peter Clark, and Christopher D Manning. Modeling biological processes for reading comprehension. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1499–1510, 2014.

[3] Vinay K Chaudhri, Britte Cheng, Adam Overtholtzer, Jeremy Roschelle, Aaron Spaulding, Peter Clark, Mark Greaves, and Dave Gunning. Inquire biology: A textbook that answers questions. *AI Magazine*, 34(3):55–72, 2013.

[4] Vinay K Chaudhri, Daniel Elenius, Sue Hinojoza, and Michael Wessel. Kb_bio_101: Content and challenges. In *Formal Ontology in Information Systems*, pages 415–420. IOS Press, 2014.

[5] Vinay K Chaudhri, Stijn Heymans, MICHAEL Wessel, and Son Cao Tran. Object-oriented knowledge bases in logic programming. In *Technical communication of international conference in logic programming*, 2013.

[6] Philipp Cimiano, Alexander Mädche, Steffen Staab, and Johanna Völker. Ontology learning. In *Handbook on ontologies*, pages 245–267. Springer, 2009.

[7] Peter Clark and Oren Etzioni. My computer is an honor student—but how intelligent is it? standardized tests as a measure of ai. *AI Magazine*, 37(1):5–12, 2016.

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[9] Andrew Gelman, John B Carlin, Hal S Stern, David B Dunson, Aki Vehtari, and Donald B Rubin. *Bayesian data analysis*. CRC press, 2013.

[10] Nikolas Gisborne and James Donaldson. Thematic roles and events. In *The Oxford Handbook of Event Structure*. Oxford University Press, 2019.

[11] Phillip J Grimaldi, Laurel Poston, and Jeffrey D Karpicke. How does creating a concept map affect item-specific encoding? *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 41(4):1049, 2015.

[12] Marti A Hearst. Automatic acquisition of hyponyms from large text corpora. In *Coling 1992 volume 2: The 15th international conference on computational linguistics*, 1992.

[13] Matthew Honnibal and Ines Montani. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. Software package, 2017.

[14] Daniel Jurafsky and James H Martin. Speech and language processing: An introduction to speech recognition, computational linguistics and natural language processing. *Upper Saddle River, NJ: Prentice Hall*, 2008.

[15] Marta M Koć-Januchta, Konrad J Schönborn, Lena AE Tibell, Vinay K Chaudhri, and H Craig Heller. Engaging with biology by asking questions: Investigating students interaction and learning with an artificial intelligence-enriched textbook. *Journal of Educational Computing Research*, 58(6):1190–1224, 2020.

[16] Marta M Koć-Januchta, Konrad J Schönborn, Lena AE Tibell, Vinay K Chaudhri, and H Craig Heller. Connecting concepts helps put main ideas together: Cognitive load and usability in learning biology with an ai-enriched textbook. *In preparation*, 2021.

[17] Xiao Ling, Peter Clark, and Daniel S Weld. Extracting meronyms for a biology knowledge base using distant supervision. In *Proceedings of the 2013 workshop on Automated knowledge base construction*, pages 7–12, 2013.

[18] Nathaniel Love and Michael Genesereth. Computational law. In *Proceedings of the 10th international conference on Artificial intelligence and law*, pages 205–209, 2005.

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

[20] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[21] Lev Ratinov and Dan Roth. Design challenges and misconceptions in named entity recognition. In *Proceedings of the thirteenth conference on computational natural language learning*, pages 147–155. Association for Computational Linguistics, 2009.

[22] Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. Snorkel: Rapid training data creation with weak supervision. *The VLDB Journal*, 29(2):709–730, 2020.

[23] Alexander Ratner, Braden Hancock, Jared Dunnmon, Frederic Sala, Shreyash Pandey, and Christopher Ré. Training complex models with multi-task weak supervision. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4763–4771, 2019.

[24] David E Sadava, David M Hillis, H Craig Heller, and May Berenbaum. *Life: the science of biology*, volume 2. Macmillan, 2009.

[25] Burr Settles. Active learning. *Synthesis lectures on artificial intelligence and machine learning*, 6(1):1–114, 2012.

[26] Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. Matching the blanks: Distributional similarity for relation learning. *arXiv preprint arXiv:1906.03158*, 2019.

[27] Hariharan Swaminathan and Janice A. Gifford. Bayesian estimation in the rasch model. *Journal of Educational Statistics*, 7(3):175–191, 1982.

[28] Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85, 2014.

[29] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771, 2019.

Appendix A. Intelligent Textbook usecase for an Accurate and Comprehensive Ontology

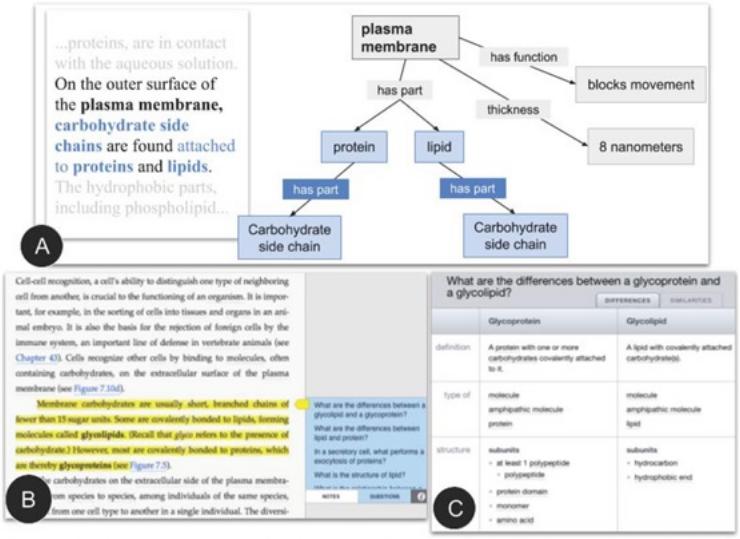

Figure 2: (A) An example sentence from the Biology textbook and its corresponding Ontology Graph. Currently, such KGs are authored manually. (B) Textbook interface for the student. Each significant word is linked to its definition and a summary page about it. In response to student highlights, the textbook automatically generates questions from the KG that a student can use for self-testing. (C) Through a structured query interface, a student can ask questions and the textbook returns the answers. In the above example, the student has asked to compare a glycoprotein with a Glycolipid.

An intelligent textbook (ITB) incorporates an ontology of key concepts and relations [3] to offer five new capabilities over and above the traditional textbooks (See Figure 2 for an illustration). First, a student can access the glossary definition of each word by simply touching on it. Second, each concept has a summary page that organizes key facts about it from across the book. This includes cross linking to different diagrams and passages across the book, and thus, breaking the linear structure of the textbook. Third, it presents the visualization of each concept that a student can interactively explore. Fourth, in response to highlights in passages, it calculates relevant questions that a student can use for self-testing. Fifth, it provides a structured query interface through which a student can pose questions to the textbook and obtain answers.

An ITB requires a comprehensive and accurate ontology graph for the following reasons. First, the ITB provides definition and visualization for every single concept, and therefore, the ontology graph must cover all salient concepts, their definitions, and properties. Second, the textbooks are expected to be nearly 100% accurate, and the students expect a similar accuracy from the anicillary resources such as test questions, visualizations, hyperlinks, etc. Finally, as an ITB is envisioned to be eventually a good tutor for the students, it must have

an accurate and comprehensive understanding of the key concepts and relationships in the textbook.

An ITB of the sort illustrated in Figure 2 has proven its usefulness to support student learning in multiple experiments. In a study with community college, the students using the ITB scored a full letter grade higher than the students using either a paper textbook or a pure textbook. None of the students in the ITB group received a D or an F, suggesting that it may be especially helpful for low performing students. A second study in a Biology classroom at Stockholm University, investigated students interaction and learning with ITB in comparison with a traditional E-book [15]. The study showed that asking questions with the AI book was associated with higher retention and correlated positively with viewing visual representations more often. In a third study at Harvard University, the students used an ITB over a period of 1.5-months in a biology classroom [16]. It was found that students displayed a beneficial learning pattern.

Appendix B. Textbook Dataset

Table 4: Textbook Dataset

Textbook	# Sentences	# Terms
OpenStax Anatomy & Physiology	21706	3196
OpenStax Astronomy	18844	810
OpenStax Biology 2e	24544	2757
OpenStax Chemistry 2e	13799	954
OpenStax Microbiology	16190	4149
OpenStax Psychology	9967	1086
OpenStax Physics Volume I	15005	462
OpenStax Physics Volume II	11779	466
OpenStax Physics Volume III	9250	580
LIFE Biology	16673	2305

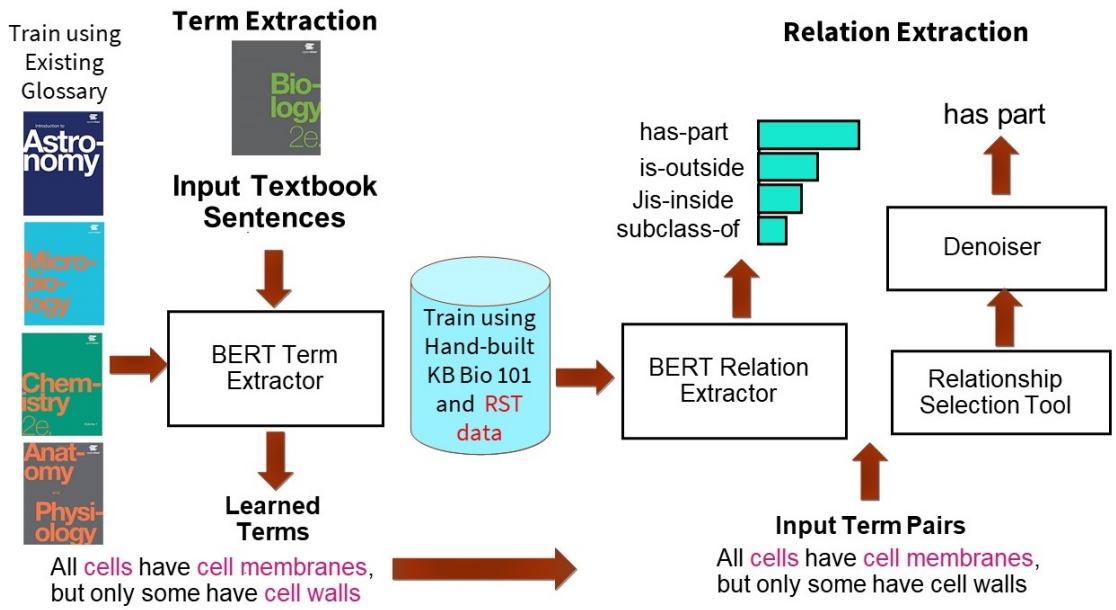

Figure 3: Illustration of the use of data in the Ontology Graph construction pipeline. Recall that RST data will be used for training the ARE in the future versions of the system.

Appendix C. Relation Selection Tool Walkthough

Relationship selection: Training

In the next portion of this study, you will be identifying how the different biology concepts you just read about are related. Before you begin, we will walk you through a brief training.

During the relationships selection task, you will be shown two terms. Your job will be to identify the relationship that exists between the terms (if there is one).

There are a LOT of possible relationships available and so, to make this easier, we have broken the task up into two parts:

- First, you will select a high-level relationship family
- Second, you will select the specific relationship within the family that you have chosen

DON'T FEEL LIKE YOU NEED TO MEMORIZE ALL THE RELATIONSHIPS! There will be reminders in the task to help you if you get stuck.

Figure 4: This is an opening screen in which the user is introduced to the task

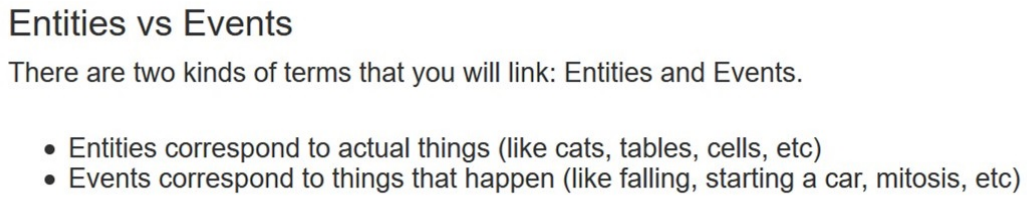

Figure 5: We first introduce the user to the distinction between an event and an entity

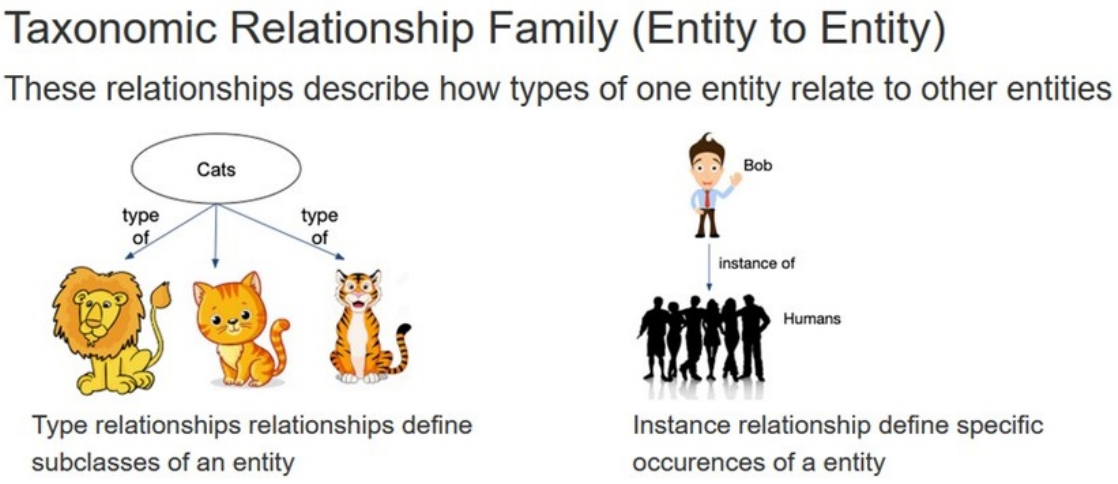

Figure 6: We teach the user about taxonomic relationships

Figure 7: This example illustrates component relationships

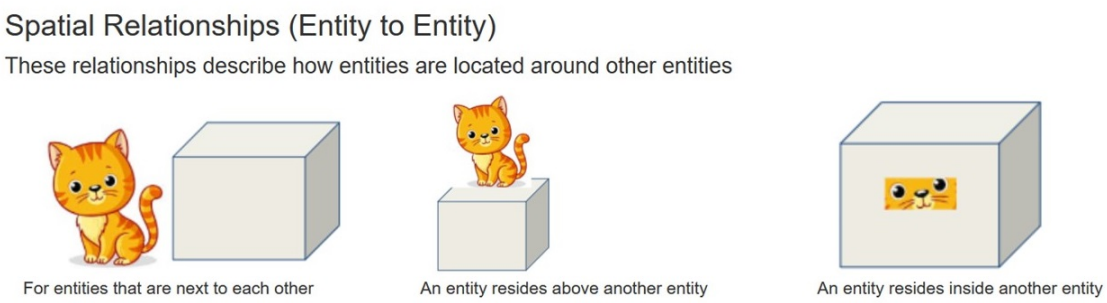

Figure 8: Example illustrations for spatial relationships

Figure 9: Examples to illustrate functional relationships

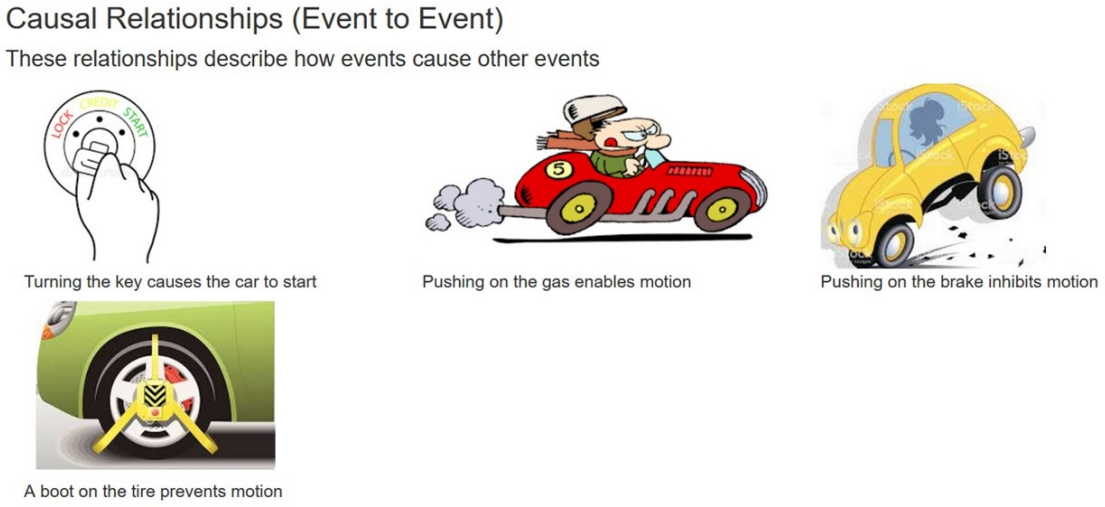

Figure 10: Examples to illustrate causal relationships

Event Structure Relationships (Event to Event)

These relationships describe when events occur with respect to other events.

Consider the example of how to boil potatoes:

Step 1: Put pot of water on stove to boil (first event)

Step 2: While the water boils, cut up the potatoes (subevent)

Step 3: After the water boils, add the potatoes (next event)

Figure 11: Examples to illustrate event structure relationships

Participant Relationships (Entity to Event)

These relationships describe how entities participate in events.

The cat pushed the vase with it's paw. The vase broke on the ground into many pieces.

The cat pushed the vase (participants)

The cat used it's paw (instrument)

The vase landed on the ground (site)

The vase broke into many pieces (result)

Figure 12: Examples to illustrate participant relationships

No Direct Relationships

Often (maybe most of the time) terms won't have a direct relationship.

Ex: The cat is on the table. The dog is outside ==> the cat and dog have no direct relationship.

Ex: The table has legs. The legs are plastic ==> the table and plastic are related through the legs, but not directly related to each other

Figure 13: Introducing the possibility of no direct relationship

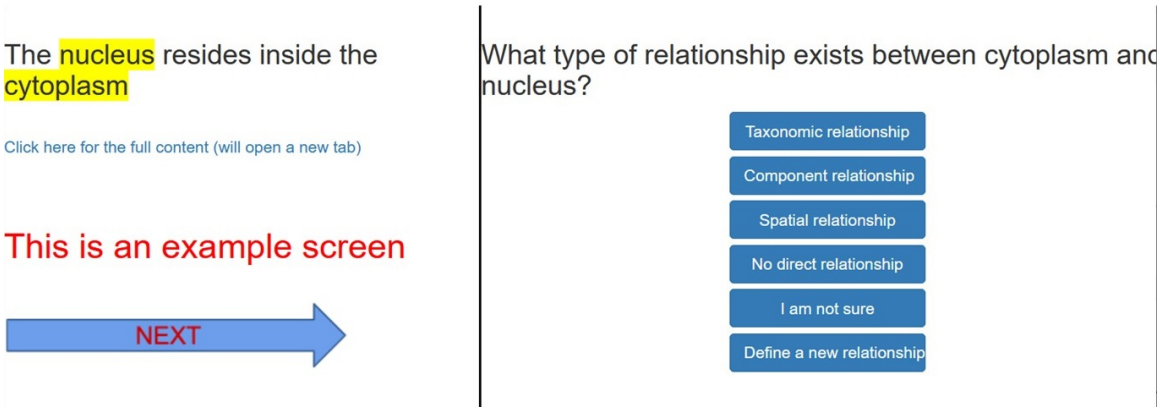

Figure 14: Start of the dialog for relation selection

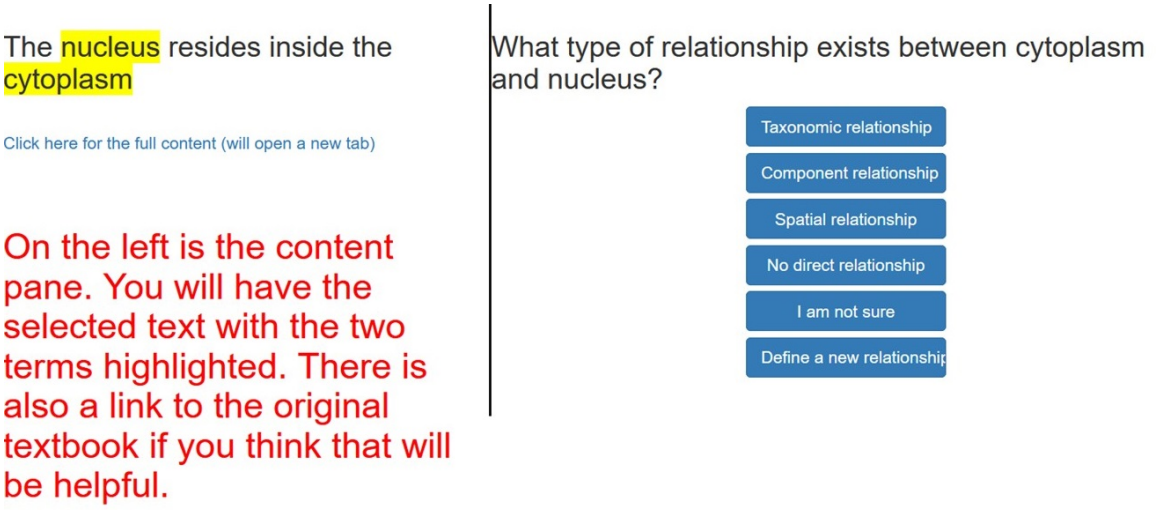

Figure 15: Explanation for the relation selection screen

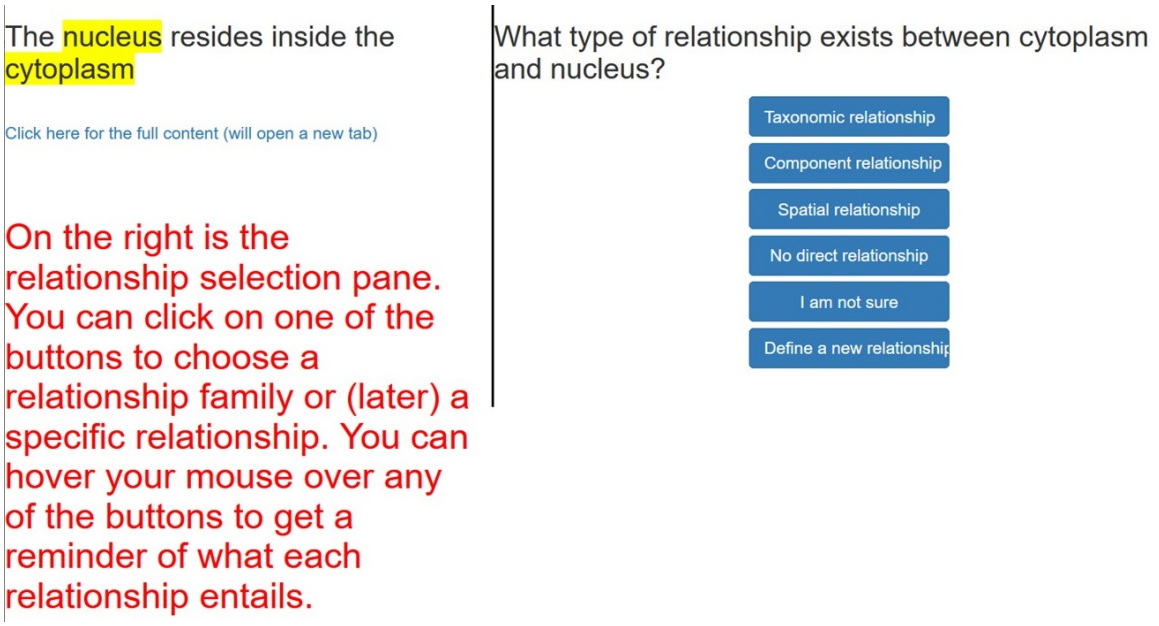

Figure 16: Continued explanation for the relation selection screen

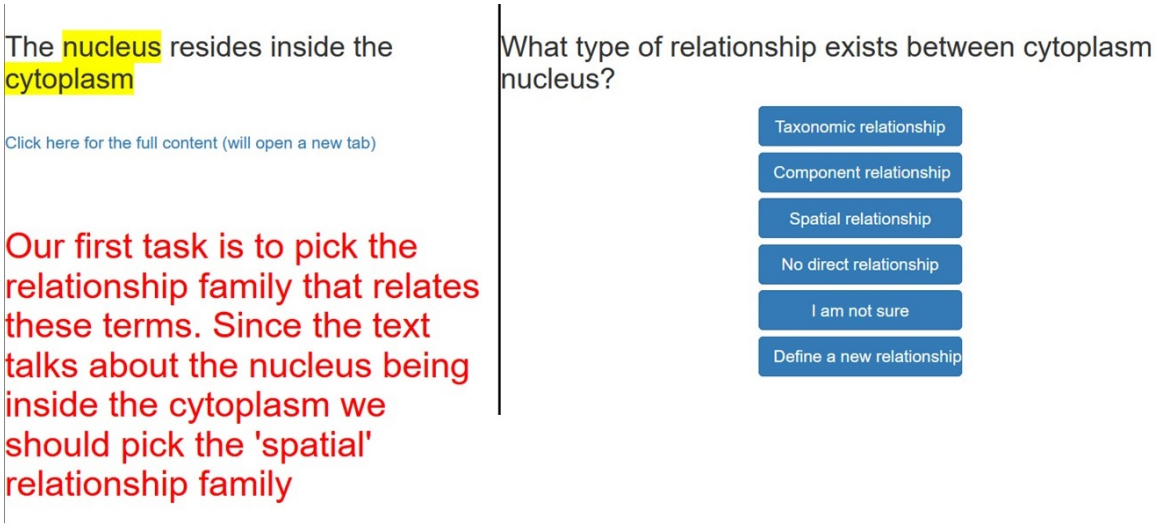

Figure 17: The user is instructed to first choose the relationship family

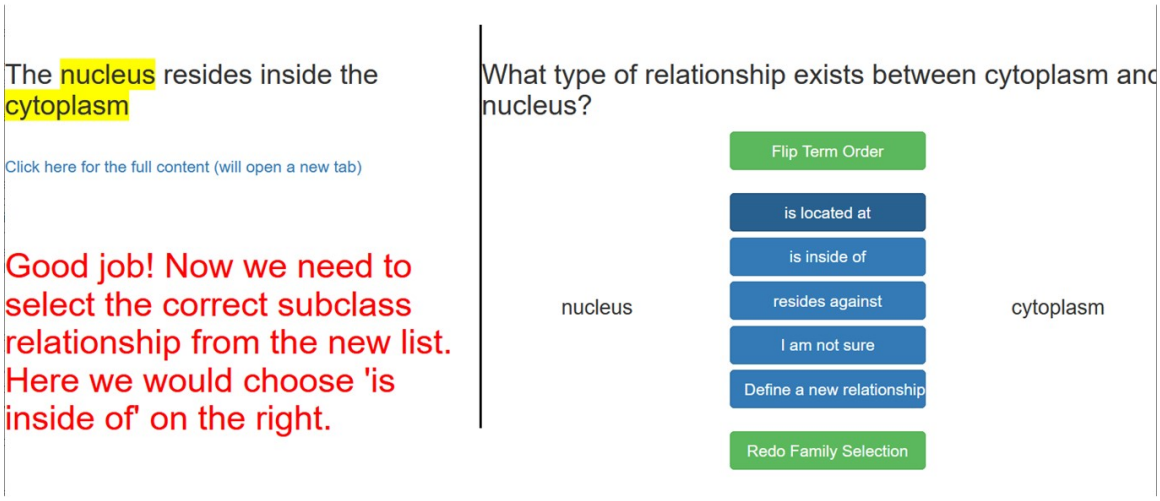

Figure 18: Once the user correctly chooses the relationship family, they are instructed to choose the relatioship

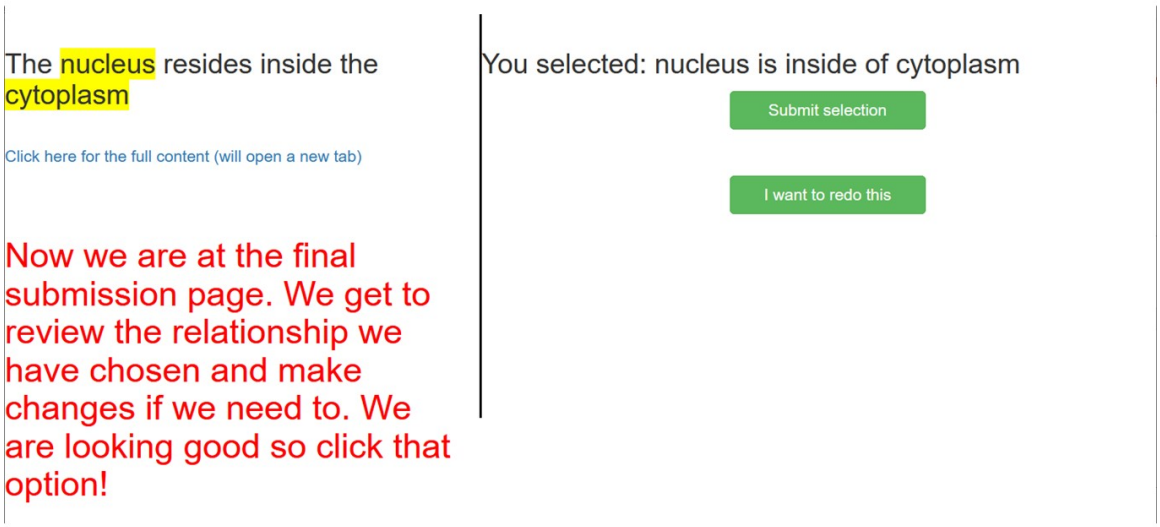

Figure 19: The user is asked to confirm the choice of relationship

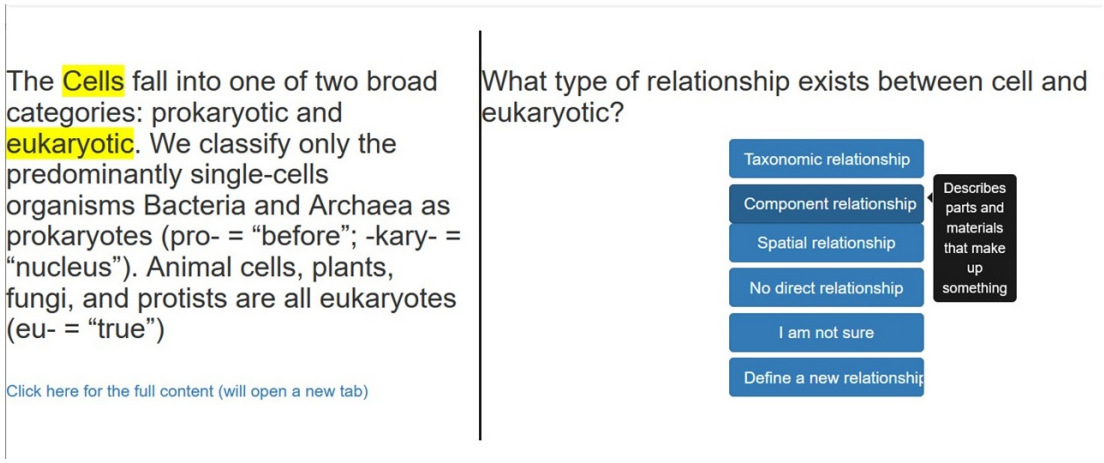

Figure 20: Another example of a relationship selection task

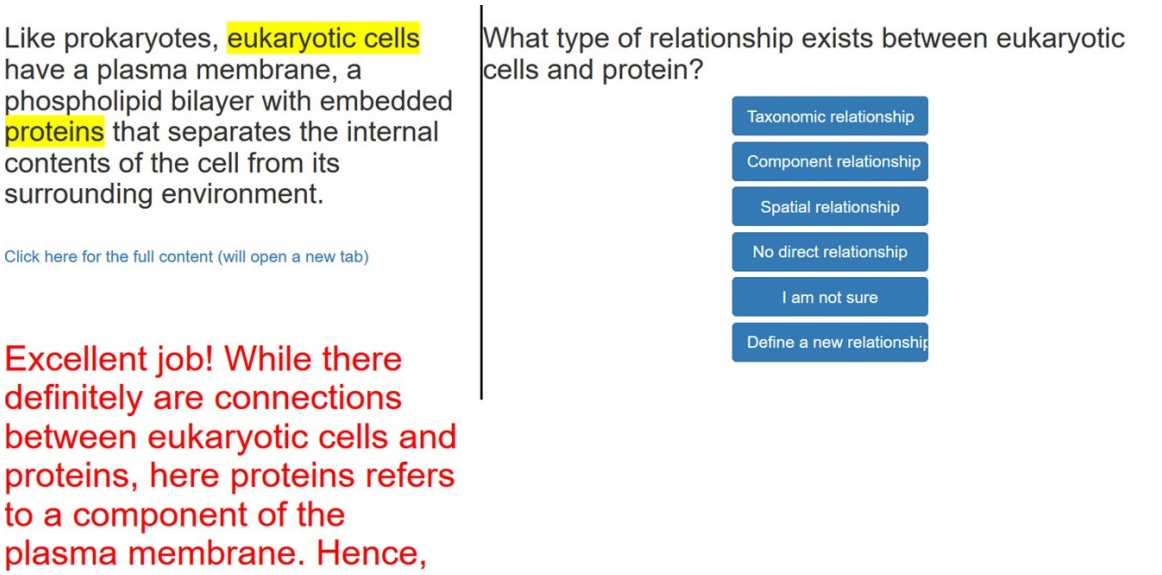

Figure 21: An example where the relationship between the terms may not be direct

Figure 22: Confirmation of the completion of the training

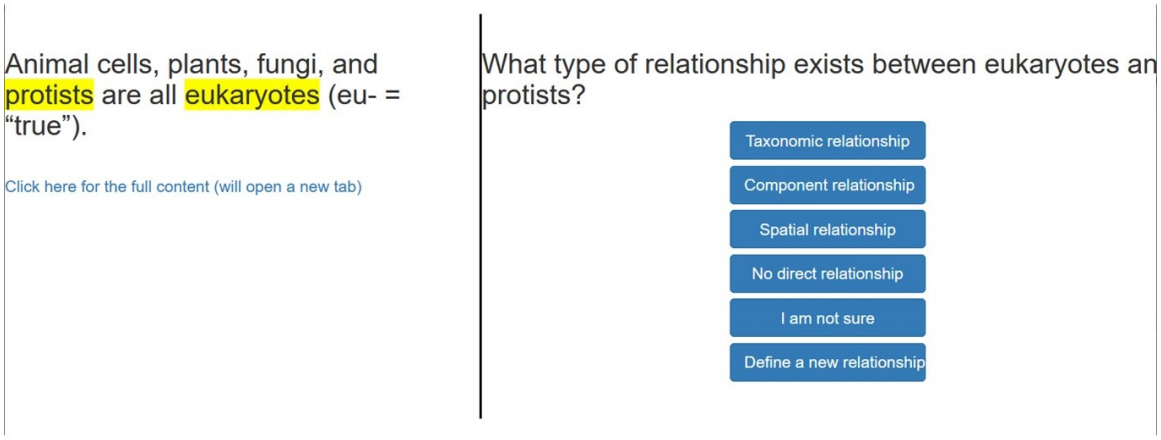

Figure 23: An example relationship selection task after the user has completed the training