
Extracting and evaluating educational concept dependencies from Large Language Models

Dominik Glandorf

Matrikelnummer 6007407

dominik.glandorf@student.uni-tuebingen.de

Anastasiia Alekseeva

Matrikelnummer 5994775

anastasiia.alekseeva@student.uni-tuebingen.de

GitHub repository: <https://github.com/mlcolab/learning-dependencies>

Abstract

1 Introduction

Large Language Models (LLMs) are trained on immense corpora of text and have proven to build on embedded factual information when performing well in downstream tasks such as question answering. Accessing this knowledge, which is represented by billions of parameters and the network's architecture, has given rise to the research field of Knowledge Extraction from LLMs (?), which rests on the assumption that the language models can help retrieve information on the relation between entities.

This work contributes to this field and aims to answer the question of whether LLMs generally are able to provide a valid dependency structure of educational concepts. To find the answer to the research question, we are going to solve two main subtasks: 1) extraction of implicit educational concept dependencies, and 2) evaluation of the retrieved dependencies or the resulting graphs by existing knowledge media.

Apart from the field of knowledge extraction from LLMs, which is a subfield of computational linguistics, our work may contribute to applied education sciences by improving computer-assisted education and creating new perspectives for a subfield of knowledge engineering that has rather stagnated in the previous years.

In particular, effective and efficient instruction does not only incorporate what to teach but also how to teach, especially the order of instruction. It is proven that *instructional sequencing* enhances the process of learning when prerequisites of educational content are known to the student or taught first [?]. Within educational content, Merrill (1983) differentiated facts, concepts, principles, rules, procedures, interpersonal skills, attitudes, and their sole recall from their application. We will focus on concepts and their relations defined understanding.

To be more precise, if one concept is a prerequisite of another, we refer to this relation as a concept dependency. For example, to understand the concept of a derivative, having knowledge about the concept of a function will facilitate or even enable learning. When the dependencies are thought of as directed edges between nodes that represent concepts, a concept dependency graph emerges which is a special type of a knowledge graph [?]. This graph is also called *concept map* in the field of Learning Sciences. The graph can be used to advance curriculum planning (?), especially for new topics that might not be covered in textbooks or automated assessment (Wang, 2015).

To conclude, our main contributions are the following. First, there is little research on how to evaluate the precision of extracted concept dependencies. Therefore, we propose a set of methods to use existing unstructured knowledge sources, namely Wikipedia and textbooks, to create baselines for evaluation. Due to the heuristic character of these methods, we conducted a manual assessment to test their suitability for our purpose. The resulting dataset can be used as a baseline for further research.

Second, the emerging field of prompt engineering provides a constellation of commands to query LLMs, the majority of which are experimental and cannot be considered fully reliable. We propose a method called *output refeeding* that allows mining the educational concept dependencies by sequentially querying the language generation model and transforming its answers into a knowledge graph.

1.1 Related work

Nguyen et al. (2011) defined the prerequisite relation in terms of the consumption of information about concepts. Vuong (2011) if a better graduation rate given prerequisite knowledge is fulfilled. Concepts are often equated with Wikipedia articles (Talukdar and Cohen, 2012; Wang, 2015).

Prerequisites can be inferred from learner behavior by testing their performance after being presented with different instructional sequences (Pavlik et al., 2008, Vuong et al., 2011). However, this has the disadvantage of disengaging users with too difficult concepts before teaching easier or necessary ones. Experts usually dispose of the required knowledge about concepts to create concept maps. The high cost of expert knowledge motivates the automated extraction of concept dependencies from appropriate sources.

Multiple approaches of prompt engineering for graph creation have been recently developed. Few-shot prompting proved successful (Wang et al., 2023).

2 Method

In this section, we will first detail our research design and then the characteristics of our information sources as well as the methods that we used to produce the knowledge graph.

2.1 Research design

The choice of methodology reflects the fact that our research is located at the intersection of educational sciences and computational linguistics. On the one hand, we want to achieve a high performance of a state-of-the-art LLM in the downstream task of knowledge graph extraction. On the other hand, we consider a complex educational problem that cannot be purely simplified to classical machine learning metrics but requires additional qualitative assessment to ensure that our results can be educationally sound.

First of all, we restricted the concepts, of which dependencies we assessed, to the domain of linear algebra. This design choice was caused by the assumptions that the highly structured domain of mathematics provides enough relations to capture on the one hand and that linear algebra is a rather matured branch of mathematics where enough data exists for our baselines to be created as well as the LLM to have learned about from its own training data. Due to the use of linear algebra in different disciplines the potential value of results is also increased.

To be able to measure the LLM’s performance, we had to come up with a target dataset. This dataset needs to describe a graph, containing the concepts as vertices and their dependencies as directed edges. Since no ground-truth, large-scale knowledge graph of concept dependencies was available to us and might even be impossible to achieve for our educational problem, we chose two sources of educational knowledge to serve as our knowledge base, namely textbooks, and Wikipedia. Due to their end-user-facing format, further processing of the input data (PDFs and webpages) was required to create the above-defined data structure. To make the graphs comparable with each other, the set of concepts was restricted to the union of all concepts indicated in all textbooks’ indices. The LLM was prompted and the responses were also further processed to create the knowledge graph.

After creating the baseline datasets and the output to evaluate, we started qualitative evaluation using an interactive dashboard, that visualized the dependency structure. This step was done to

get an impression of the general suitability of all three pipelines and identify potential reasons for performance losses. Then a subset of all three datasets was manually rated by multiple raters to not only infer their overall quality but also assess the human performance on the problem. We finally checked for outputs for convergence to give suggestions for a scalable, low-cost evaluation system of future (versions of) LLMs. The former evaluations are rather typical in educational research whereas the latter enables the creation of large-scale datasets, potentially suitable to train and fine-tune machine learning models.

2.2 Baseline extraction

This section details how we used Wikipedia and textbooks to set up the evaluation baselines for the LLM.

2.2.1 Wikipedia

The choice of the online encyclopedia Wikipedia as an educational information source is a result of its inherent graph structure induced by hyperlinks between articles and empirical evidence of its suitability for this purpose. Especially, the platform is claimed to cover concepts up to the complexity of undergraduate curricula to a high extent, which aligns well with our chosen domain of linear algebra, which is usually taught to undergraduate students. The focus on dependencies required for understanding a concept makes an encyclopedia an ideal source because its articles are usually written to enable a fast understanding of the article’s topic. In an online encyclopedia, hyperlinks provide additional information. In Wikipedia, each content article begins with a short summary, often serving as a definition and embedding the topic into its context, before in-depth information follows. In this summary text, we assume an implicit order from most important to least important concepts that are used to explain the topic without digressing to other concepts or unimportant information. The involved concepts are inferred to be a dependency. To prevent direct cycles by two concepts involving each other in their explanation, the one that refers earlier to the other is considered to be dependent on the latter. Situations, where the introduction of concept A states that concept B is dependent on it might be resolved like this.

Our assumptions about the Wikipedia structure were operationalized in the following way to compute the dependencies for a given concept. Every link in the first content paragraph containing a link is considered a candidate dependency. Therefore, any non-content paragraph such as advice regarding irrelevance or missing sources has to be removed beforehand. First, links to categories or fields such as "Mathematics" or "Linear Algebra" are removed based on a manually composed blacklist (a more sophisticated, automatic pruning could make use of Wikipedia’s category pages). Then links to articles about persons are filtered out since these do not represent educational concepts. The same applies to links to pages that contain unexpected content such as disambiguation pages. Subsequently, cyclic dependencies are resolved as described above using both character positions of the symmetric links in the text. If more candidates remain than the threshold of 5, only the first 5 were kept, using the assumption of decreasing importance from the beginning on and the constraint of some sparsity of the resulting graph. This set of concepts is then taken as the concept’s dependencies.

The described procedure was executed for every concept in the set resulting from the textbook baseline. Although Wikipedia theoretically offers a larger number of articles about the topic, we decided to stick with the limited set to maintain comparability. However, we decided against restricting the dependencies to other concepts in the set to also be able to evaluate the recall of the methods. The computational complexity of the procedure is relatively low and most time is spent on requesting Wikipedia web pages for the concepts and the links. The overall time was hence reduced by caching the responses to these requests. The technical implementation in Python required engineering our own Wikipedia API using the Python libraries requests and BeautifulSoup. The tools are used to download the page content and conveniently access elements such as links or special HTML tags indicating that an article is about a person.

2.2.2 Textbooks

We chose student textbooks as the complementary information source besides the general purpose knowledge base Wikipedia because they are written by experts specifically for the purpose of educational knowledge transfer to learners. This does not only lead to comprehensive coverage of

carefully selected concepts that are considered relevant by the authors but also to a specific structure that we planned to exploit with our baseline extraction method, namely a pedagogic linearization of topics. More precisely, the authors are expected to first introduce concepts that other more complex concepts build upon. Another useful structure in textbooks is the index, which provides a comprehensive list of concepts defined as its own entities and covered in the book, and indicates the area of introduction.

Due to the popularity of the topic of Linear Algebra, we could easily identify ten free textbooks on the topic and obtain their PDF representations (either directly typeset in this format or scanned and processed with optical character recognition). The books comprised 584 pages on average (652,297 characters) and disposed of a median of 382 index entries.

First, the textbooks were converted from PDF format to unformatted text. To identify all concepts of interest covered in a book, the pages containing the index were identified and each index entry with its corresponding pages was extracted in a semi-manual fashion. This required setting parameters such as the title, position of the text columns, and the types of nesting and delimiters. To enable comparing the knowledge graph across the baselines, we decided to use the set of Wikipedia articles as the set of possible concepts, meaning that only a Wikipedia article title can serve as a vertex in the knowledge graph. This required to map from each index entry to the closest Wikipedia article. For this disambiguation, the index entry was introduced into the Wikipedia search engine suffixed with the field term "(Mathematics)" to hint at the general area. Due to the high diversity of the resulting ten articles, their titles were string-matched with the search term (excluding the suffix) using the Levenshtein distance (provided by a Python library). The article title with the lowest Levenshtein distance that did not refer to a disambiguation page was the result of this so-called *Wikisearch disambiguation*.

For dependency extraction, we came up with two distinct strategies. The first is solely based on indices whereas the latter also uses the full text. The first strategy is called *order pruning* and is focused on unlikely dependencies on the graph. It checks whether two concepts are introduced at least twice in a specific order and rules out the dependency that is not suggested by this order. For example, if concept A is introduced twice before concept B, it is unlikely to depend on a concept introduced later by two independent authors. To create a sparse graph, this strategy requires a massive amount of books, which was not available to us, but it can help to prune dependencies and refine a candidate graph.

This candidate graph is the output of the second heuristic called *common introductory usage*. This is based on a similar rationale as the Wikipedia heuristic because it figures out which other concepts are used to introduce a new concept. The first and rather structured source of information is the index, again. Some books list multiple pages for index entries, indicating that the concept occurs again at a later point in a significant way. If concept A appears for the second time on the first page for concept B, A is likely to be a dependency of B. However, the structured source of index entries has a low amount of data, which led to the involvement of the full text. Here, entity recognition plays a crucial role because not every word is a concept of interest and not every concept occurs with its canonical title in natural text.

One convenient method is provided by the Wikifier [?]. The underlying algorithm looks for links in the entire Wikipedia with the link text matching a word in the input. Naturally, the same text can link to distinct articles, which requires disambiguation based on the context. Wikifier solves this by creating a graph based on all potential links and uses a PageRank algorithm to calculate the score for each potential link target. We restricted links to correspond to a link text of at least 3 characters to get rid of abbreviations that were contained in formulas randomly. We used a minimum PageRank of 0.00005 as a threshold after manual inspection aimed at a good recall of links but also good precision. The Wikifier was called via its online API. All detected link targets on the first page of an index entry serve as candidate dependencies which are then ranked by the following three criteria: First, the number of books in which it occurs in the area of introduction. Second, the average number of occurrences in these books. Third, the PageRank of the link. As with the Wikipedia dependencies, only the 5 highest-ranking candidates are saved as dependencies in the knowledge graph. This procedure was carried out for all index entries across all books.

The final resulting graph is the union of dependencies identified by the two common introductory usage sources (indices and full text), pruned by the unlikely dependencies from order pruning.

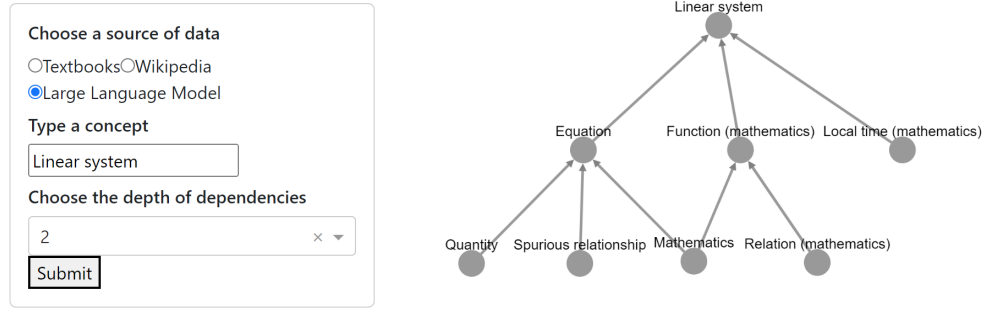


Figure 1: A screenshot of the interactive application.

2.3 LLM extraction

Due to the closed-source character of the very powerful and successful language model GPT3 by openai and the unavailability of an API for its most recent application ChatGPT, we decided to use the open-source model BLOOM by bigscience, which is a joint effort by hundreds of researchers around the world and is based on the same transformers architecture as GPT3 and has a comparable amount of parameters. Because of resource limitations and our specific use case, we used a derived model called *T0pp* (<https://huggingface.co/bigscience/T0pp>), which only has 11 billion parameters and is fine-tuned for zero-shot task generalization, especially question-answering. The dataset used for training T0pp is also available under the name P3 (Public Pool of Prompts) (<https://huggingface.co/datasets/bigscience/P3>). We set up the language model on the cloud infrastructure of the university's excellence cluster and ran it in generation mode (no training) with a maximum response length of 50 tokens on a node with eight cores of the type Nvidia GeForce RTX 2080 Ti. This model consumed approximately 44GB of RAM and a typical request took three to ten seconds to be answered completely. To save time loading the model into the memory, we set up an HTTP server to conveniently query the model via GET requests from our local machines.

Manual trial and error testing of our chosen and other LLMs (such as meta's Galactica) has led to the following procedure to extract dependencies from the knowledge base of T0pp, which we call *output refeeding*. It is based on the previously explained assumption that understanding a concept requires understanding other concepts involved in its definition. Given a concept of interest, we therefore first prompt the model for a definition of the concept: "What is the mathematical definition of concept?" The response is defined as definition and involved in the second prompt: "What mathematical concepts are mentioned here: definition". The model usually responds with a comma-separated list that is split by commas and then disambiguated in the same way as the index entries in the textbook baseline by our Wikisearch disambiguation. Finally, self-dependencies, that might occur during this procedure, are deleted from the set of dependencies. This procedure is run for the same set of concepts as the Wikipedia baseline.

2.4 Manual inspection via Dashboard

As indicated in our design, this educational problem requires also a qualitative evaluation. We hence developed a Dashboard via graphing library for Python Plotly¹ (Fig.1). The application is devised so that one could examine the resulting graph itself in a convenient environment. It allows plotting the concept dependencies graphs for three knowledge sources and types of dependency mining methods (LLM, Wikipedia, and textbooks) for different depths of dependencies, meaning that dependencies of dependencies are shown. The dashboard also has potential use for educators that are interested in concept dependencies.

¹<https://plotly.com/python/>

2.5 Manual baseline

The novel character of our baselines and the resulting uncertainty about their appropriateness to evaluate the LLM’s performance motivated us to evaluate all three knowledge graphs independently before checking their convergence. Therefore, 4 raters were presented with 100 dependencies from each graph and had to rate them in a binary fashion (correct vs incorrect). We only evaluated the quality of dependencies for concepts that occurred at least in 50% of the textbooks to ensure that there is enough data for the textbook baseline and that we assess the data for important concepts only. For the ratings, we calculated Cohen’s Kappa as a measure of interrater reliability. This should not only guarantee the confidence of the performance evaluation but also assess the difficulty of the problems for human raters.

2.6 Convergence statistics

3 Results

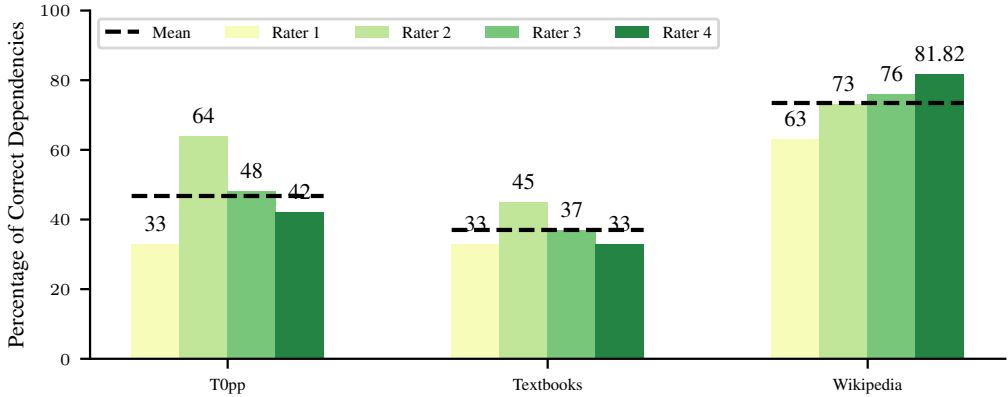


Figure 2: Rating.

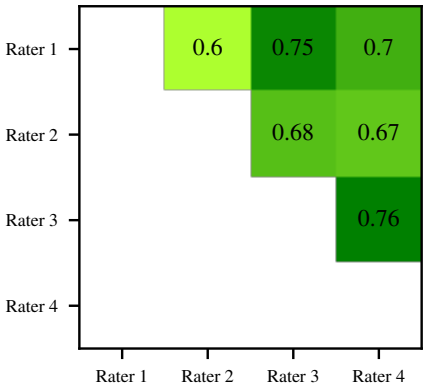


Figure 3: Interrater reliability, kappa coefficient.

4 Discussion

Each information extraction method is not fully independent of the other, in particular, entity recognition and concept disambiguation steps in the procedure for LLM and textbooks depend on Wikipedia.

As shown, current LLMs are not sufficiently precise in creating concept maps, which confirms previous research ?. This could be improved by training the pre-trained models on the knowledge

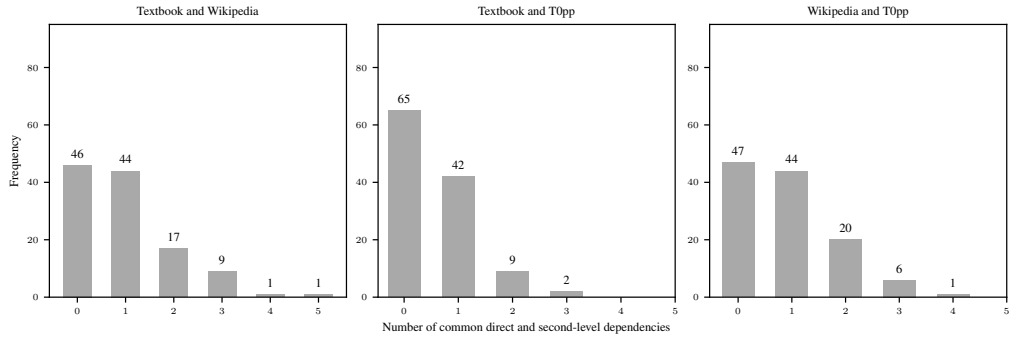


Figure 4: Convergence metrics. Number of common direct dependencies between sources.

graph data mined from Wikipedia or any other reliable sources of concept dependencies in a specific subject of instruction ?.

5 Acknowledgements

We sincerely thank our project supervisors, Álvaro Tejero-Cantero and Hanqi Zhou.

6 Appendix

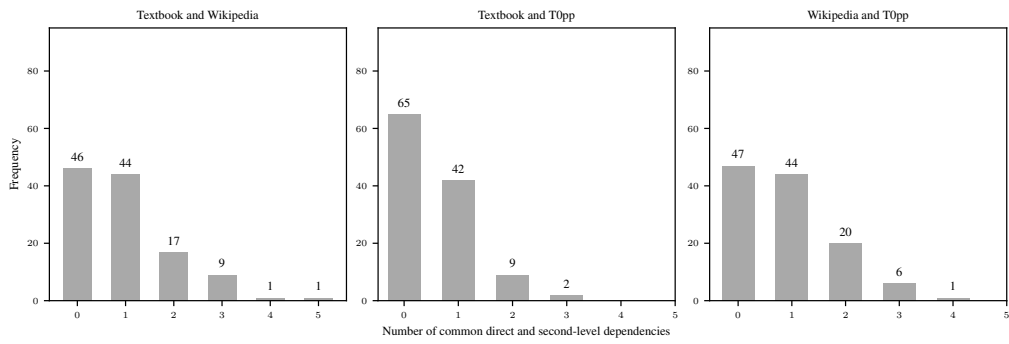


Figure 5: Convergence metrics. Number of common direct and second-level dependencies between sources.