Symbol Description Reading

Karol Lynch[1], **Bradley Eck**[1], **Joern Ploennigs**[1,2]

[1]IBM Research Europe, Dublin, Ireland
[2]University of Rostock, Rostock, Germany
{karol_lynch,bradley.eck,joern.ploennigs1}@ie.ibm.com, joern.ploennigs@uni-rostock.de

Abstract

Mathematical formulas give concise representations of a document's key ideas in many natural sciences and engineering domains. The symbols that make up formulas carry semantic meaning that may differ by document or equation. What does x mean in a given paper? Interpreting the symbols that comprise formulas requires identifying descriptions from the surrounding text. We approach this task of symbol description reading as an application of current AI technologies targeting the tuning of large language models for particular domains and automation of machine learning. Our pipeline integrates AI question answering and natural language processing to read symbol descriptions. We consider extractive and generative AI model variations and apply our pipeline on two example tasks of symbol description reading. Promising results provide motivation for wider deployment for which we describe a microservice architecture and related challenges.

Problem Introduction & Significance

Recent foundation models like GPT4 show astonishing capabilities to understand natural language and encourage the development of Large Language Models (LLM) that specialize on knowledge engineering and natural sciences to assist practitioners in those fields with models of deep domain understanding. But, knowledge in those domains is often formally modelled in mathematical formulas that the LLMs need to understand. Currently LLMs capture the text around the formulas, but, do not embed the knowledge within the formulas (Lin et al. 2019). Two challenges currently limit the reach of LLMs to mathematical formulas. First, formulas need to be parsed and interpreted structurally rather then sequentially. Second, symbols used in the formula should be linked to their definition in the text in order to embed their meaning in the LLM. Libraries exist for the parsing task (like *texvc*), but description reading remains a challenging problem with few examples of successful applications. It is the main topic of this paper.

Symbol description reading (SDR) is the task of reading descriptions of mathematical symbols from unstructured text like Wikipedia articles or scientific papers. As an example of how formulas often appear in documents, consider the following presentation of mass-energy equivalence[1]:

> The formula $E = mc^2$ defines the energy E of a particle in its rest frame as the product of mass (m) with the speed of light squared (c^2).

The formula is an easy case as it includes the description of all identifiers E, m, and c in the text after the formula. The objective of the SDR task is to read these symbol definitions from the article.

For us, SDR is an important application of AI technology with many novel uses such as accessibility of mathematical content to the visually impaired (Karshmer, Gupta, and Pontelli 2007), information retrieval (Schubotz et al. 2017; Alexeeva et al. 2020), automated machine learning (Ba et al. 2023), and embedding formula knowledge in LLMs. Creating a deployable solution in practise requires the integration of methods from different areas of AI including question-answering, machine reading comprehension and generative models into one integrated AI pipeline. We present such an AI pipeline using a question answering (QA) approach and evaluate it on two distinct but related tasks and show successful results on this interesting application.

The remainder of the paper is organized as follows. First we review the relevance of AI for the problem and highlight the opportunity for improvement with a novel integration of existing methods. Next we present our pipeline including extractive and generative QA variations. Then we realize the pipelines for two related, but distinct, tasks and datasets. We conclude with considerations for deploying SDR in enterprise systems providing automatic machine learning and domain specific fine-tuning of LLMs.

Relevance of AI Technology and Related Work

Many scientists and practitioners in the fields of natural and engineering science are looking with huge interest at the possibilities of LLMs to retrieve, summarize and explain the large body of work that exists in those fields. Many topics in those areas are captured in formulas that need to be embedded into the LLMs to model and explain their meaning.

Related SDR Tasks

Several different tasks related to mathematical information extraction have been studied in the past. The first approaches

[1]https://w.wiki/d7M, 15.11.2023

were looking into (i) extracting descriptions of entire formulas (Nghiem Quoc et al. 2010). Descriptions at the formula level do not however consider individual symbols and their meaning. Therefore, the next approaches looked into (ii) extracting descriptions of mathematical expressions within formulas (Kristianto, Topic, and Aizawa 2014; Lin et al. 2019). But, to fully understand the formulas it is needed to (iii) extract the descriptions of the contained mathematical symbols, which is the SDR task. SDR approaches actually group into two variants: (iii.A) Mathematical Identifier Description Reading (**MIDR**) which is the extraction of symbol definitions at document level (Schubotz et al. 2016, 2017; Lynch, Ploennigs, and Eck 2023; Alexeeva et al. 2020). Another approach (iii.B) Symbol extraction and description linking (**Symlink**) operates at the passage level, extracting the symbol and its definition from one or a few given paragraphs in the context of the formula (Lee and Na 2022; Popovic, Laurito, and Färber 2022; van der Goot 2022).

SDR tasks are challenging for several reasons. Symbol descriptions are often not contiguous and so cannot be fully solved by extracting a single string from the document. Moreover, symbols usually occur many times in a document (24.2 on average in the MFQuAD dataset (Lynch, Ploennigs, and Eck 2023)) and only some—or even none—are associated with an explicit description (Wolska and Grigore 2010).

Existing Approaches

For the MIDR task, the best performing approaches in the literature use hybrid noun phrase ranking (NPR) as proposed by Schubotz et al. (2017). An evaluation of Schubotz et al. (2017)'s approach on the *MFQuAD dataset* in Lynch, Ploennigs, and Eck (2023) reported an Exact Match (EM) score of 0.326 and an F1 score of 0.402. An improved version of NPR by Lynch, Ploennigs, and Eck (2023) achieves an EM score of 0.477 and an F1 score of 0.529 on the *MFQuAD* dataset.

For the Symlink task, numerous approaches were introduced at SemEval 2022 under task 12. Several participating systems (Lee and Na 2022; Popovic, Laurito, and Färber 2022; van der Goot 2022) were summarized in Lai et al. (2022). The winning system, JBNU, by Lee and Na (2022) used separate models for named entity recognition and relation extraction. They employ SciBERT, fine-tuned separately for each task, as an encoder for both models.

The innovation in our SDR pipeline is applying a question answering technique using LLMs to the problem.

Question Answering for SDR

Our approach to SDR is informed by the following observations. First, pre-trained machine reading comprehension models, such as *RoBERTa* (Liu et al. 2019), that are fine-tuned for question answering are available. Second, queries for an identifier's description can be easily formed from templatised natural language questions. Finally, an identifier's description almost always occurs close to an occurrence of the identifier. This proximity means a full passage of the document can be sent to the reading comprehension model for processing. For example, in the MFQuAD dataset 97.5 % of descriptions occur within 165 characters of their identifier (Lynch, Ploennigs, and Eck 2023), well within the 512 token input width of *RoBERTa*.

Hence, we propose a pipeline for symbol description reading applying a question answering approach with four steps as described below:

1. *Passage retrieval* returns the context or passage surrounding the appearance of a target symbol.
2. *Passage pre-processing* replaces formulas and symbols with tokens more useful for a machine reading comprehension model.
3. *QA Answer scoring* considers each span of the passage as an answer to the question "What is the definition of `<identifier>`?" and returns a confidence score.
4. *Answer re-ranking* aggregates confidence scores across multiple occurrences of an identifier.

Passage retrieval involves returning a context centred around each target identifier occurrence. Identifier occurrences are located using a regular expression. The regex considers various representations of mathematical identifiers that may be present in the document such as HTML, Unicode, or LaTeX. A context, delimited by sentence boundaries, is then extracted from the characters around each identifier occurrence. We use NLTK's[2] implementation of the Kiss and Strunk (2006) sentence splitter, including a custom version trained on mathematical source documents. Configuration parameters for passage retrieval include the margin width and tokenizer for sentence boundaries.

Passage Pre-processing replaces formulas and symbol definitions with tokens a reading comprehension model can understand. Pre-processing also normalizes identifier encodings. We evaluate three strategies for the representation of mathematical expressions and formulas in the final surface text: (i) using a unicode representation of the entire formula where possible, otherwise the raw LaTeX representation (UNICODE); (ii) using the unicode representation of the left hand side (LHS) of a formula; (iii) or replacing formulas with semantic tags, an approach used in Schubotz et al. (2017) (SEMANTIC). We replaced expressions with *expression*N and formulas with *formula*N, where N denotes its occurrence order in its source article. Finally, different encodings of the same identifier (e. g. HTML, Unicode, or LaTeX[5]) are normalised to one unicode representation.

The *QA Answer Scoring* step uses a reading comprehension model to evaluate answers to the question "What is the definition of `<identifier>`?". We pass this question to the model together with the pre-processed contexts for the identifier. The model scores each span in the identifier context, based on its confidence that it is a correct answer and a description of the identifier for the paired query question. The model also outputs a NULL answer score, representing its confidence that the passage contains no description for the target identifier. This QA step—with and without fine tuning for mathematical identifiers—represents the key innovation of our approach hence we explore extractive and generative model variations as further described below.

[2]https://www.nltk.org/

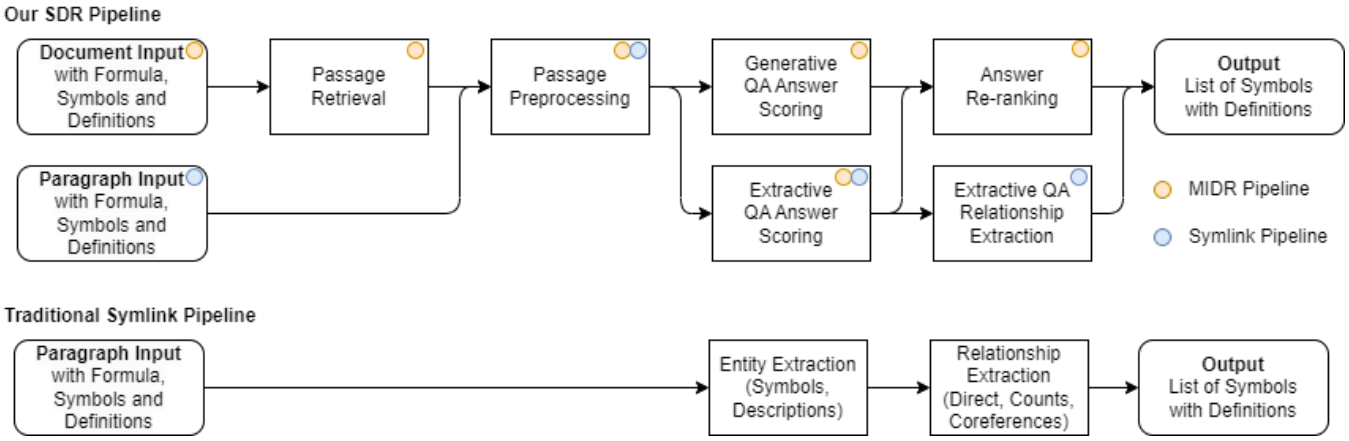

Figure 1: Extractive and Generative QA Approach in comparison to traditional Symlink Approach

As the final step in the pipeline, *answer re-ranking* evaluates the multiple candidate descriptions that may be found for an identifier to identify the best description. To compare answer scores across passages we follow the approach of Kratzwald, Eigenmann, and Feuerriegel (2019) and train a re-ranking model on a combination of retrieval and reading comprehension features. Our implementation differs in using non-neural models and a different feature set. Our re-ranking features are: (i) the order of occurrence of the target identifier in the source article, (ii) the top answer score for each passage, (iii) the NULL answer score for each passage, (iv) the top answer score normalised by the NULL answer score ($ln(e^{\text{top\_score}}/e^{\text{null\_score}})$), (v) the rank of this passage based on feature (ii), and, (vi) the rank of this passage based on feature (iv). By re-ranking scores for answers from different parts of the document, we aim to find the best description for a symbol, considering the whole document.

Figure 1 summarizes the steps of our pipeline, including extractive and generative variations. The figure also identifies, by colored circles, applicable blocks for the MIDR and Symlink tasks. The variations to our SDR pipeline are described in the following subsections.

Extractive QA for MIDR

In applying our SDR pipeline to the MIDR task with an extractive model, we follow the four steps in our pipeline and use an encoder architecture and span classification head in the QA Answer Scoring step (Figure 2). This architecture replicates that used by Devlin et al. (2019) for extractive question-answering tasks like SQuADv2. A span classification head, consisting of a span start linear layer and a span end linear layer, is added on top of the hidden-states output by the base encoder model (i.e., SciBert or RoBERTa). It computes the span start and end logits for the query answer (i.e., the target identifier description). The score of each span in the context is the sum of the start logit of its first token and the end logit of its last token. The NULL score (i.e. no description of the target identifier in the context) equals the sum of the start and end logits of the CLS token.

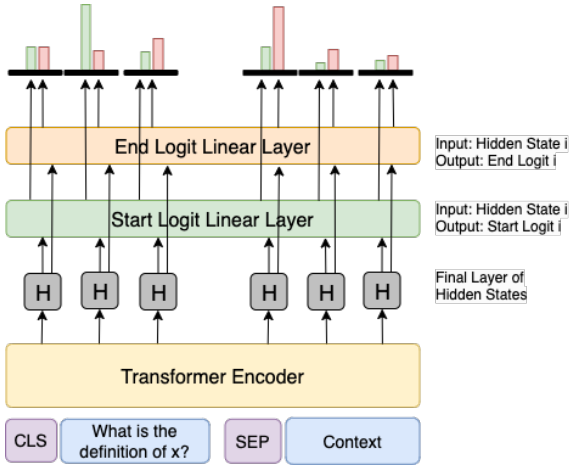

Figure 2: Architecture of extractive answer scoring model.

Generative QA for MIDR

The generative QA approach to the MIDR task uses the same high level steps as the extractive QA approach: passage retrieval, passage preprocessing, answer scoring and answer re-ranking (c.f., Figure 1). Only the details of the answer scoring step differ with the extractive approach. In place of extractive reading comprehension models (an encoder architecture and a span classification head), the generative approach uses the text-to-text T5 model (Raffel et al. 2020) which has an encoder-decoder architecture with a language modelling head. T5 was trained on multiple downstream NLP tasks, including QA, by adding task-specific markers to the original input. We evaluate several variants of the T5 model with and without fine-tuning, maintaining the same markers used to train each T5 variant for QA tasks.

In the answer scoring step we assemble a model input (i.e., a prompt) from a retrieved passage containing a target identifier and a question asking the model to generate a description of the target identifier. Then, just like the extractive approach, we calculate a score both for the top candidate

description, and the models confidence that there is no description for the target identifier in the passage (both these scores form part of the input to the answer re-ranker). Beam search is used to select the top answer candidate. If there is no description for an identifier within a passage, the model should indicate this by generating a specific output text (e.g. "unanswerable") that is configured during general QA fine-tuning. We use the generative models' next token probabilities to generate both confidence scores (normalized by their length): one for the top candidate answer and one for there being no definition.

Extractive QA for Symlink

For the Symlink task, only passage level annotations are provided by the dataset, thus passage retrieval and answer re-ranking across passages are not required. Moreover, this task is purely extractive (all entities are spans of the context); thus we apply a variation of our extractive SDR pipeline. We utilise a two step approach, with symbols being extracted in the first step, and symbol descriptions, counts and their relations being extracted in the second step. We use the NER approach of Lee and Na (2022) for symbol extraction. Reading descriptions of the symbols follows the same QA answer scoring method as the extractive MIDR variation. We again pass the question "What is the definition of <*identifier*>?" together with the passage, but we query for relations associated with each identifier occurrence separately with distinct representations of each target identifier occurrence. To facilitate a more complete comparison of our Symlink results, we additionally use the query "What is the count of <*identifier*>?" for extracting *Count* relationships.

Evaluation

MIDR Task Evaluation

We evaluated the performance of extractive and generative variations of our pipeline for the MIDR task using the Math Formula Question Answering Dataset (MFQuAD) (Lynch, Ploennigs, and Eck 2023). This dataset provides over 7500 annotated occurrences of 310 mathematical identifiers. These examples include many tricky aspects of MIDR such as identifiers only described implicitly or not at all. The categories are:

- *All* - all 310 identifiers in the MFQuAD dataset.
- *Explicit* - 175 identifiers with at least one explicit description.
- *Implicit* - 49 identifiers with only an implicit description.
- *Some* - 224 identifiers with explicit or implicit description (Union of *Explicit & Implicit*).
- *Missing* - 86 identifiers without explicit or implicit description in the article.

Our evaluation on MIDR aimed at finding good configuration options for the pipeline, at investigating performance of fine-tuning for the QA approach, and at comparing our extractive and generative pipelines to the existing results. For configuration, we test minimum context margin sizes of 150, 200 and 250 characters, breaking at the first sentence boundaries outside these cutoffs. We also consider the three mathematical object preprocessing strategies: UNICODE, LHS, and SEMANTIC. Further, we evaluate both the default and custom sentence splitters. For answer reranking we assessed the following models: (i) random forest (RF) with 12 estimators and max tree depth of 6; (ii) Logistic Regression (LR) with an L2 penalty term and C=1.0; (iii) SVM with $\gamma = n_{features}^{-1}$, kernel='rbf', C=1.0. With regard to performance comparisons, we report the metrics Exact Match (EM) and F1-score as used in the official SQuAD scoring script (Rajpurkar, Jia, and Liang 2018). Exact match is the fraction of examples for which the description matches exactly. The F1-score gives a balanced measure of overall model performance, capturing the model's skill on positive examples (recall) and its accuracy on the examples it captures (precision). For fine-tuning experiments, we select a subset of the best pipeline configuration options. All evaluations used 5-fold cross-validation with consistent splits across approaches.

Extractive QA Approach For the extractive MIDR pipeline we first studied configuration options and then the effect of fine-tuning specifically for MIDR.

We evaluated all configuration options using both *SciBERT* and *RoBERTa* models fine-tuned solely on the SQuADv2 dataset. We show the top performing configurations in Table 1. All top configurations used a margin width of 150 and LHS pre-processing. The results show that *SciBERT* is outperforming *RoBERTa* in most cases. For the sentence splitter, both the default and our custom splitter perform well in different cases. The Logistic Regression (LR) model showed the best re-ranking performance.

To evaluate the effect of fine-tuning for MIDR, we carried out an additional step of fine tuning using the MFQuAD dataset. We used the AdamW optimiser with a learning rate of $5e^{-5}$, a batch size of 16 and trained for either 1, 2 or 4 epochs. The loss function is the mean of the cross entropy loss of the span start and end classifier. Top results for the extractive pipeline with MIDR fine tuning appear in Table 2. Fine-tuning improves both models and makes the performance of *RoBERTa* much closer to that of *SciBERTA*. Fine tuned results showed less depencence on the re-ranking model, with all three configurations appearing in the top results. Overall, it is clear that fine-tuning improves the EM and F1 results significantly. Considering all identifers, the best exact match score rose from 0.416 to 0.671, an improvement of 61%. Similar uplift is observed for F1, rising from 0.450 to 0.716 or 59%. Although fine-tuning gave improvements for all groupings, performance remains low in absolute terms for *Implicit* identifiers highlighting the opportunity for generative methods.

Generative QA Approach We evaluate the generative QA approach with several variants of the text-to-text T5 model (Raffel et al. 2020): t5-small, t5-base, t5-small-squad2neg. Answer generation uses beam search with 2 beams, α set to 0.0 or 0.6 and max output sequence length set to either 16 or 32. Consistent with the extractive approach, MFQuAD finetuning is evaluated only for the pre-

ID Group	Model	Sentence Splitter	Re-ranking Model	Exact Match	F1
All	SciBERT	Cus	LR	0.416	0.450
	SciBERT	Def	LR	0.416	0.446
Explicit	SciBERT	Def	LR	0.417	0.495
	RoBERTa	Cus	RF	0.406	0.481
Implicit	SciBERT	Def	SVM	0.041	0.115
	SciBERT	Def	LR	0.041	0.115
Some	SciBERT	Def	LR	0.335	0.410
	SciBERT	Cus	LR	0.321	0.398

Table 1: Top extractive MIDR results without fine-tuning.

ID Group	Model	Epochs	Re-ranking Model	Exact Match	F1
All	RoBERTa	2	LR	0.671	0.716
	RoBERTa	2	SVM	0.668	0.712
Explicit	SciBERT	1	SVM	0.771	0.831
	RoBERTa	2	LR	0.754	0.815
Implicit	RoBERTa	4	RF	0.061	0.191
	SciBERT	2	LR	0.041	0.258
Some	SciBERT	1	SVM	0.612	0.699
	SciBERT	1	RF	0.585	0.688

Table 2: Top extractive MIDR results with fine-tuning.

processing configurations that perform best when no fine-tuning is used.

We study all generative models with and without fine-tuning on the MFQuAD dataset, with the top results for each model shown in Table 4. Again finetuning the models significantly improves the performance. Without finetuning the *t5-small-squad2neg* model is in many scenarios the best choice, with finetuning the *unifiedqa-v2* is the best performing model. In most cases LHS is the best preprocessing strategy in combination with re-ranking using LR.

MIDR Results Discussion A comparison between the top extractive QA, generative QA and the state of the art Noun Phrase Ranking (NPR) approach from (Lynch, Ploennigs, and Eck 2023) that is an improved version of Schubotz et al. (2017) is shown in Table 3. It's clear the two QA approaches are generally more effective than the NPR approach for the MIDR task. Between the two QA approaches the extractive one achieves slightly better results across *All* identifiers.

ID Group	Approach	Exact Match	F1
All	Extractive QA	**0.671**	**0.716**
	Generative QA	0.655	0.694
	NPR	0.477	0.529
Explicit	Extractive QA	**0.771**	**0.831**
	Generative QA	0.771	0.821
	NPR	0.491	0.625
Implicit	Generative QA	**0.224**	**0.393**
	Extractive QA	0.061	0.191
	NPR	0.000	0.150
Missing	Generative QA	0.744	0.744
	Extractive QA	**0.872**	**0.872**
	NPR	0.802	0.802
Some	Generative QA	**0.652**	**0.728**
	Extractive QA	0.612	0.699
	NPR	0.384	0.521

Table 3: The top results of each MIDR approach.

Both the QA approaches have a similar level of performance for *Explicit* defined identifiers. Correctly identifying when a context does not contain the answer to a question, form an important part of a readers performance. Whilst the extractive QA approach was the most effective for this category of *Missing* definitions, the generative QA approach was the least effective, even performing less well than the NPR which uses a much simpler model. The hardest task are *Implicit* descriptions. Here generative QA approach performs significantly better than the extractive approach. Particularly, as it manages to answer many implicit examples that require an ability to read basic mathematical syntax.

Overall, the QA approach shows a significant improvement after MIDR specific finetuning of the base models. We conjecture two reasons for this. First, the QA dataset that both models were trained on, the SQuADv2 dataset, does not contain mathematical content. Second, the MIDR task uses the same question template for all examples, so the fine-tuned models can adapt to the question, as well as the content.

Symlink Task Evaluation

We finally evaluate our QA approach for the Symlink task using the Symlink dataset[3]. Here we investigate if our QA approach also works well on a different task and dataset. Considering the uplift due to fine-tuning observed on MIDR, we also fine-tune here.

The Symlink dataset contains predefined *train*, *dev*, and *test* splits, which we use for model training, hyperparameter selection, and validation, respectively. We implement the same superset of the metrics defined in Task 12 at SemEval 2022 to evaluate the named entity recognition and the relation extraction. The first is using the F1 score metrics for partial, type and strict entity matches, which requires an exact entity character span and type match. The relation extraction uses the standard precision, recall, F1 score metrics, with a candidate relation correct iff it strictly matches the gold standard. We report relation extraction results for both strict entity matches and overlapping (spans overlap) entity matches. The results are presented in Table 5 and compared against the SOTA approach JBNU of Lee and Na (2022), which was the leading submission at SemEval 2022.

We fine-tune the *SciBERT* model using the AdamW optimiser with a learning_rate of 5e-5, for 1 epoch with a batch size of 16. The context passed to the model consists of 150 characters of text, either side of the target symbol (300 plus the symbol length characters in total). During evaluation, we execute two separate queries for each symbol, a query for *Direct* relations, and a query for *Count* relations, then select the better query result between the two. The query that returns the highest ratio of top span score, to

[3]https://github.com/laiviet/symlink_modeling/tree/main/data

ID Group	Generative Model	Tuned	Margin	Pre-proccesor	Sent. Split	Re-rank. Model	α	MaxLen	Epochs	Exact Match	F1
All	t5-small-squad2neg	No	250	SEM	Cus	SVM	0.0	32	-	0.394	0.417
	unifiedqa-v2	Yes	250	LHS	Def	LR	0.6	32	4	0.655	0.694
Explicit	t5-base	No	150	UNI	Def	LR	0.0	32	-	0.383	0.438
	unifiedqa-v2	Yes	250	LHS	Def	LR	0.6	32	4	0.771	0.821
Implicit	t5-small-squad2neg	No	200	LHS	Cus	RF	0.6	16	-	0.041	0.161
	unifiedqa-v2	Yes	250	LHS	Def	LR	0.6	32	4	0.224	0.393
Some	unifiedqa-v2	No	250	LHS	Def	LR	0.0	32	-	0.304	0.414
	unifiedqa-v2	Yes	250	LHS	Def	LR	0.6	32	4	0.652	0.728

Table 4: The top results of generative pipeline for MIDR with and without fine-tuning.

	Entity F1 Scores			Relation (Strict Entities)			Relation (Partial Entities)			Relation (Type Entities)		
System	Strict	Partial	Type	Prec	Recall	F1	Prec	Recall	F1	Prec	Recall	F1
Extractive QA	**0.4633**	**0.5444**	**0.5327**	**0.3580**	**0.3024**	**0.3279**	0.5397	**0.4580**	**0.4955**	0.5139	**0.5528**	**0.5327**
JBNU	0.4499	0.5378	0.5274	0.3038	0.2426	0.2698	**0.5662**	0.4397	0.4950	**0.5233**	0.5316	0.5274

Table 5: Results for the Symlink task. We only consdier Direct and Count relations for the RE task.

NULL answer score (i.e., $e^{span\_score}/e^{null\_score}$) is considered the better result. In other words, we normalise the span scores using the NULL answer scores, before comparing the two query results. Using the results from the top query, we infer a relationship if the difference between the span score and the NULL answer score is greater than a configurable answer_threshold (i.e., $span\_score - null\_score > answer\_threshold$), and no relationship otherwise. The *answer_threshold* hyperparameter is configured using the *dev* split, with 4 returning the best results.

Symlink Results Discussion The results in Table 5 show that our extractive QA approach is effective for the Symlink task. We observe an improvement on the existing result for both studied sub-tasks. For strict entity recognition, our F1 score of 0.4633 improves the prior value by 3% and for strict relation extraction our F1 of 0.3279 represents an improvement of 21%. It works for both *Count* as well as *Direct* relation extraction. The results also provide evidence that our extractive QA approach holds up when complex LaTeX symbols are text are fed directly to the model. On average, symbol entities in Symlink have more characters (8.6) than MIDR symbols (4.1). Moreover, Symlink symbols include complex expressions such as summations.

Deployment Considerations

Our SDR approach is implemented in a microservice architecture where we have different services for document parsing depending on whether we have LaTeX(*texvc*), HTML (Lynch, Ploennigs, and Eck 2023), or PDF (Livathinos et al. 2021) documents; the presented QA pipeline service; and web services to store and serve the results.

We use the microservice in different applications. One is the automatic configuration of time series forecasting models in IoT for which the modelling of causalities of underlying laws of physics is relevant. Here we use the SDR service to parse relevant scientific papers to extract targeted symbols and definitions. We map those into a semantic knowledge graph of an IoT system that we then transpose into a Heterogenous Graph Neural Network to train a prediction model (Ba et al. 2023). For this use case, exact matches between definitions and the existing semantic graph is of high relevance to capture the right physical causalities.

A further application area enabled by this contribution is creating domain specific LLMs in the engineering and natural science domain. This requires not just extracting individual identifiers and definitions from scientific papers, but, embedding a formula's internal relations into the LLM. Larger collections of documents are needed to extract generalizable domain knowledge, where repeatable matches of similar identifiers is more relevant. The larger document base also renders GPU requirements an important constraint, making smaller model variants that run on CPU more practically relevant for scalability.

Conclusions

In this paper we presented the SDR task as an application of AI question answering technology. We found that applying QA boosted performance by more than 59 % on MIDR and on the Symlink task more than 1.3 % on named entity recognition and 21% on strict relation extraction. Further, we show that fine tuning can give models such as T5-base an exciting ability to read mathematical notation interspersed with natural language. We also show that the generative T5-base model, with its consistent text-to-text format, struggles to identify when an example does not contain an answer to a question, performing worse on this class of examples than much simpler models.

The recent advances in LLMs and their emerging applications across industries renders the embedding of the content of formulas highly relevant to fully unlock their potential in engineering and natural science domains. Solving the SDR task is key, and applying LLM advances in QA approaches and generative models. Then we will be able to create domain specific LLMs for math, physics, engineering, and commerce that actually have mathematical and causal understanding within, to interpret, reason and support users in their various tasks.

References

Alexeeva, M.; Sharp, R.; Valenzuela-Escárcega, M. A.; Kadowaki, J.; Pyarelal, A.; and Morrison, C. 2020. MathAlign: Linking Formula Identifiers to their Contextual Natural Language Descriptions. In *Language Resources and Evaluation Conference*, 2204–2212.

Ba, A.; Lynch, K.; Ploennigs, J.; Schaper, B.; Lohse, C.; and Lorenzi, F. 2023. Automated Configuration of Heterogeneous Graph Neural Networks With a Semantic Math Parser for IoT Systems. *IEEE Internet of Things Journal*, 10(2): 1042–1052.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*, 4171–4186.

Karshmer, A.; Gupta, G.; and Pontelli, E. 2007. Mathematics and accessibility: A survey. In *Int. Conf. on Computers Helping People with Special Needs*, volume 3118, 664–669.

Kiss, T.; and Strunk, J. 2006. Unsupervised Multilingual Sentence Boundary Detection. *Computational Linguistics*, 32(4): 485–525.

Kratzwald, B.; Eigenmann, A.; and Feuerriegel, S. 2019. RankQA: Neural Question Answering with Answer Re-Ranking. In *ACL*, 6076–6085.

Kristianto, G. Y.; Topic, G.; and Aizawa, A. 2014. Extracting Textual Descriptions of Mathematical Expressions in Scientific Papers. *D Lib Mag.*, 20.

Lai, V.; Pouran Ben Veyseh, A.; Dernoncourt, F.; and Nguyen, T. 2022. SemEval 2022 Task 12: Symlink - Linking Mathematical Symbols to their Descriptions. In *Int. Workshop on Semantic Evaluation (SemEval)*, 1671–1678.

Lee, S.-M.; and Na, S.-H. 2022. JBNU-CCLab at SemEval-2022 Task 12: Machine Reading Comprehension and Span Pair Classification for Linking Mathematical Symbols to Their Descriptions. In *Int. Workshop on Semantic Evaluation (SemEval)*, 1679–1686.

Lin, J.; Wang, X.; Wang, Z.; Beyette, D.; and Liu, J.-C. 2019. Prediction of Mathematical Expression Declarations Based on Spatial, Semantic, and Syntactic Analysis. In *ACM Symposium on Document Engineering 2019*.

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR*, abs/1907.11692.

Livathinos, N.; Berrospi, C.; Lysak, M.; Kuropiatnyk, V.; Nassar, A.; Carvalho, A.; Dolfi, M.; Auer, C.; Dinkla, K.; and Staar, P. 2021. Robust PDF Document Conversion Using Recurrent Neural Networks. In *AAAI*, 17, 15137–15145.

Lynch, K.; Ploennigs, J.; and Eck, B. 2023. What Makes a Good Dataset for Symbol Description Reading? arXiv:2304.08352.

Nghiem Quoc, M.; Yokoi, K.; Matsubayashi, Y.; and Aizawa, A. 2010. Mining Coreference Relations between Formulas and Text using Wikipedia. In *Workshop on NLP Challenges in the Information Explosion Era (NLPIX)*, 69–74. Beijing, China: Coling 2010 Organizing Committee.

Popovic, N.; Laurito, W.; and Färber, M. 2022. AIFB-WebScience at SemEval-2022 Task 12: Relation Extraction First - Using Relation Extraction to Identify Entities. In *Int. Workshop on Semantic Evaluation (SemEval)*, 1687–1694.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1): 5485–5551.

Rajpurkar, P.; Jia, R.; and Liang, P. 2018. Know What You Don't Know: Unanswerable Questions for SQuAD. In *ACL*, 784–789.

Schubotz, M.; Grigorev, A.; Leich, M.; Cohl, H. S.; Meuschke, N.; Gipp, B.; Youssef, A. S.; and Markl, V. 2016. Semantification of Identifiers in Mathematics for Better Math Information Retrieval. In *SIGIR*, 135–144.

Schubotz, M.; Krämer, L.; Meuschke, N.; Hamborg, F.; and Gipp, B. 2017. Evaluating and Improving the Extraction of Mathematical Identifier Definitions. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction*, 82–94.

van der Goot, R. 2022. MaChAmp at SemEval-2022 Tasks 2, 3, 4, 6, 10, 11, and 12: Multi-task Multi-lingual Learning for a Pre-selected Set of Semantic Datasets. In *Int. Workshop on Semantic Evaluation (SemEval)*, 1695–1703.

Wolska, M.; and Grigore, M. 2010. Symbol Declarations in Mathematical Writing. In *Towards a Digital Mathematics Library. Paris, France, July 7-8th, 2010*, 119–127.