Datasets Search Engine

E. Ferizolli, K. Johannes, M. Khovansky, A. Wang

February 2025

Contents

1 Introduction

Finding relevant research datasets remains a significant challenge due to inconsistent metadata and suboptimal retrieval methods [1, 2], as well as limited query processing approaches [1, 2, 3]. OpenML's diverse dataset collection, for instance, often lacks uniform descriptions, making it challenging to capture both the broad and nuanced characteristics of data. Even recent efforts such as DataFinder [4], which employ neural search methods, fall short by relying on a single textual representation of datasets and limited training data. Hence the aim of this paper is to help researchers find datasets relevant to their research goals.

To this end, in order to address the limitations described above, we introduce a multi-component document representation approach, leveraging dataset title, description, and a large language model (LLM)-generated summary of feature/column names to fully encapsulate dataset properties. Additionally, we expand the scope of training data by aggregating more datasets than were explored in previous neural search approaches. We also utilise newer, more advanced language models for better natural language comprehension and introduce a more complex scoring function that's more suitable for multi-component document representation.

The datasets we use to create our document collection are from the OpenML API [5] and DataFinder dataset [4]. Our information retrieval method searches through a collection of 5981 verified datasets

from OpenML, and an additional 7000 datasets from DataFinder. This is usually done by utilising some metadata, but in our research, we extract metadata about each dataset and then create additional metadata to make the search more informed. In particular, we extract the dataset unique identifier (ID), title, description, feature names, and publication year. In addition, we apply an LLM to create a summary of the feature names that is supposed to describe the dataset in a more informative way.

We provide an effective information retrieval system to help researchers find the most relevant datasets. We complete this task by breaking it down into more modular and manageable subtasks. These subtasks include document creation, dataset augmentation, retraining of a more advanced model, query enrichment, and benchmarking our implemented improvements.

2 Review of Related Works

Due to a lack of metadata standardisation, early approaches to dataset search had to rely on work-arounds to traditional information retrieval methods; among such work-arounds were filtering or classification. For example, Kunze et al's search engine retrieved relevant datasets based on topical, geographical, temporal, and other filters [1], while Lue et al used classification for keyword tagging [3].

Recent developments in metadata standardisation have enabled newer approaches to utilise traditional information retrieval methods such as BM25 [6, 4]. A significant development took place in 2020, when Brickley et al from Google introduced the Dataset Search tool – a search engine designed to index and retrieve datasets across the web, encompassing approximately 45 million datasets from 13,000 distinct sources [7]. The system operates as a metadata-centric search engine, leveraging an open ecosystem to scale to the entirety of metadata published on the web, and representing a pioneering metadata standardisation effort.

Building on Brickley et al's efforts, Viswanathan et al in 2023 introduced DataFinder: a novel neural search algorithm that uses the vector representations of bi-encoders fine-tuned on query-dataset pairs to compute semantic similarity between queries and the textual representations of datasets; the queries were generated using a generative model (Galactica [8]) based on the abstracts of dataset-using papers, and they were matched with relevant datasets using a rule-based approach [4]. The neural search significantly outperformed traditional search engines on their test data, which was human-labelled. However, the team reported that the scope of their dataset collection was limited, and that their query collection was restricted to the English language; moreover, they reported that natural-language queries were not fully exploited [4].

Viswanathan et al's neural search approach was enabled by earlier advancements in natural language processing (NLP). Devlin et al from Google in 2018 introduced BERT, a large language model (LLM) based on encoder-only transformer architecture – introduced earlier in 2017 [9, 10]. By incorporating context surrounding each token using the attention mechanism, BERT was able to generate context-aware word embeddings, as well as sentence-level embeddings using the CLS token. This represented a major milestone in NLP and made meaningful sentence-level semantic matching possible [9].

DataFinder used a version of BERT trained on scientific text – SciBERT – to generate its bi-encoder vector representations. SciBERT is a pre-trained language model based on BERT designed to tackle the lack of high-quality, large-scale labeled scientific data [11]. The model uses unsupervised pre-training on a large multi-domain corpus of scientific publications. This has been shown to improve performance on downstream scientific NLP tasks such as dataset search engines. This work focuses on the abilities of unsupervised pre-training of language models on large corpora, returning contextualised embeddings for each token, which can be passed into minimal task-specific neural architectures. SciBERT showed a significant improvement compared to the BERT base model across many domains, such as biomedical and computer science. It also achieved new SOTA results on many tasks such as NER and CLS. SciBERT researchers found that BERT fine-tuning yielded superior results compared to task-specific architectures atop frozen embeddings [11] .

Recent developments in the field of science-oriented LLMs have highlighted the potential to address the latter limitation identified by Viswanathan et al: the inefficient extraction of information from

natural-language queries. Specifically, SPECTER2 was introduced in 2023, and is the current SOTA model for learned representations of scientific documents [12, 13]. Building upon the fact that these can be valuable input features for downstream tasks without further fine-tuning, it addresses the difficulty in capturing the diversity of relevant tasks. SPCTER2 improves the generalisation ability of scientific document representation models by learning multiple embeddings per document, each tailored to a different format. Using task-format-specific control codes and adapters, Singh et al create a family of multi-format models. The work builds on the success of multi-task learning, exploring it in the realm of scientific document representation, and optimising a suitable objective for task formats. SPECTER2 achieves an accuracy of 62.9, which includes adapters and multi-task learning control codes in-train, 74.1 out-of-train, and 71.2 on average [12, 13].

Another limitation identified by Viswanathan et al was the narrow scope of their dataset collection. We intend to address this limitation via OpenML, an open-source platform for machine learning researchers to share and organise data in fine detail [5]. It employed networked science tools that allowed scientists to make discoveries much faster, building directly on observations of all others. OpenML was introduces as an online service to share, organise and reuse data, code and experiments, allowing collaborations to scale easily and reward scientists for sharing their data openly. Specifically related to datasets, what is accepted includes only a limited number of formats. This creates a standardised collection of information. An array of data characteristics is computed, including measures like number of features, instances, classes, missing values, statistical and information-theoretic measures, and landmarkers [5].

In summary, the limitations identified by Viswanathan et al concerning natural language query processing and dataset collection scope expose a gap in the literature. We intend to close this gap by using DataFinder as a baseline but expanding the dataset collection using OpenML and improving natural language query processing by using a newer bi-encoder (based on SPECTER2). We will also introduce additional adjustments and test a number of modifications, as detailed in section *3 Architecture*.

3 Architecture

The dataset search engine is structured into multiple interconnected modules responsbile for OpenML document creation, training set generation, fine-tuning of SPECTER2, scoring function definition, and evaluation.

3.1 High-level architecture

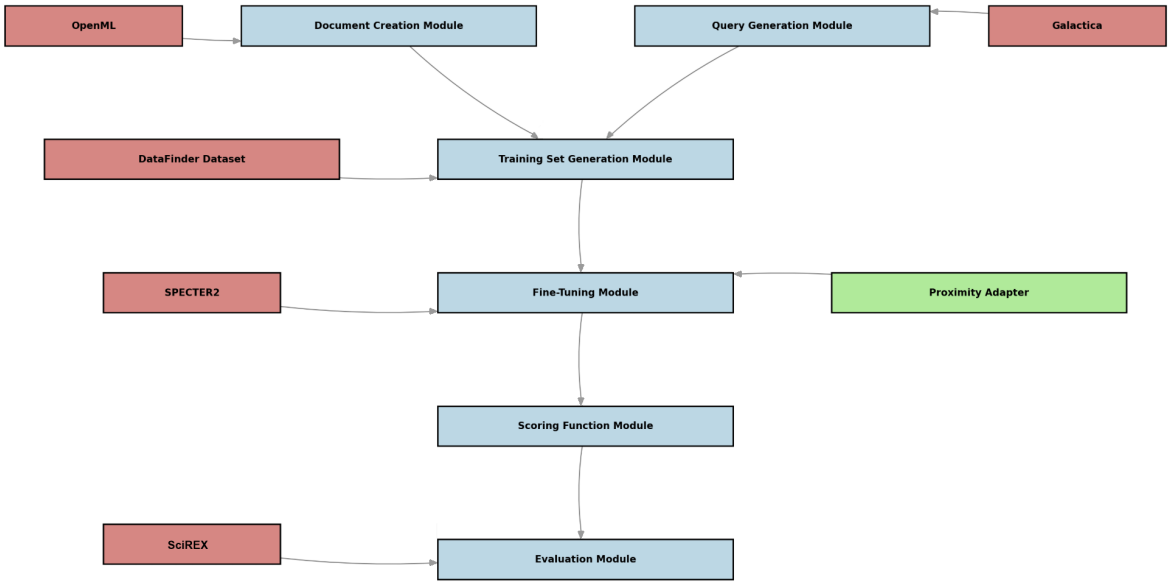

Figure 1: High-level architecture diagram

3.2 Document Creation Module

The process begins with the document creation module, where documents are generated by extracting metadata from datasets retrieved via OpenML, which includes the dataset title, publication year, and feature names (schema labels). To make effective use of feature names, we will use Galactica – a large language model fine-tuned on scientific text [8] – or a newer LLM (e.g. Gemini) to generate condensed summaries of the feature names using few-shot learning on manually created examples. The metadata are then concatenated and stored in a structured JSON format. The final document representation is created by generating a list of three components: dataset title, dataset description merged with publication year, and feature name summary. A series of preprocessing steps will be applied to ensure the data is clean and ready for use.

3.3 Query Generation Module

Following the methodology of Viswanathan et al, training queries are generated using Galactica, which is asked to generate five keyphrases: the tasks mentioned, the scientific domain, the modality of data, the language of data or labels, and the length of text. However, since OpenML does not provide the abstracts of dataset-using papers, we will generate the queries from dataset descriptions rather than paper abstracts.

3.4 Training Set Generation Module

Once the documents and queries are ready, the training set is generated by first creating query-document pairs from the prior two modules, and then merging them with the training set provided by Viswanathan et al (converted into our 3-part document representation format detailed in subsection *3.2 Document Creation Module*, resulting in an expanded dataset. The expanded dataset is then split into training and validation subsets to avoid data leakage upon further testing (section *4 Experimental Design and Testing Plan*).

3.5 Fine-Tuning Module

The fine-tuning module is responsible for training the bi-encoder, SPECTER2, using contrastive loss between the positives and hard negatives. The positives will be all the query-document pairs in the training set, while the hard negatives will be generated by applying BM25 to the queries and selected those documents that BM25 ranks highly but that are not labeled as relevant in the training set. To enhance the fine-tuning process, a custom proximity adapter will be developed, which allows the model to focus on semantic similarity.

3.6 Scoring Function Module

To define the scoring function, the model computes SPECTER2 representations for both the query and the document. As described in subsection *3.2 Document Creation Module*, each document is composed of three key components: the title, the extended description, and the feature name summary. The CLS (Classification Token) representations of both the query and document are extracted to generate similarity scores, with the similarity function then being defined as the dot product between the CLS representats of the query and document:

$$\text{sim}(q, d_{\text{component}}) = \text{CLS}(\text{SPECTER2}(q))^T \cdot \text{CLS}(\text{SPECTER2}(d_{\text{component}}))$$

Instead of relying on a simple similarity function as per Viswanathan et al, we introduce a weighted sum of the similarity function's output when applied to the 3 components that each document comprises (title, extended description, feature name summary), where the weights are to be optimised using hill climbing on the validation set:

$$\text{score} = \sum_{d_{\text{component}} \in D} \text{sim}(q, d_{\text{component}})$$

,

where D is the set of the 3 components comprising each document's representation.

This approach allows the search engine to factor in specific contents of the dataset rather than just its general description.

3.7 Evaluation Module

The final step in the architecture is the evaluation module, which measures the search engine's effectiveness using the four key metrics used by Viswanathan et al, allowing for direct comparison:

1. Precision@K is defined as the fraction of the top K retrieved datasets that are relevant. This metric emphasizes early precision, measuring how many of the highest-ranked results are correct. It is crucial in scientific search scenarios where researchers often examine only the top few results.

$$P@K = \frac{\text{\# of relevant datasets among top } K}{K} \tag{1}$$

2. Recall@K quantifies the fraction of the total relevant datasets that appear in the top K. When a researcher is looking for a dataset, multiple datasets may be relevant for a single query, making it essential to measure how thoroughly the system uncovers all the relevant datasets.

$$R@K = \frac{\text{\# of relevant datasets among top } K}{\text{total \# of relevant datasets}} \tag{2}$$

3. Mean Average Precision (MAP) measures the overall ranking performance by considering the position of each relevant dataset in the results list. For a single query q, the Average Precision (AP) is computed as:

$$AP(q) = \frac{1}{R} \sum_{k=1}^{N} (P@K \times \mathbb{1}(\text{dataset at rank } k \text{ is relevant})) \tag{3}$$

where R is the total number of relevant datasets for query q, and $\mathbb{1}(\cdot)$ is an indicator function equal to 1 if the dataset at rank k is relevant. MAP is then computed as the average of $AP(q)$ over all queries:

$$MAP = \frac{1}{|Q|} \sum_{q \in Q} AP(q) \tag{4}$$

MAP rewards systems that rank relevant datasets higher and consistently retrieve them at optimal positions across all queries. It penalizes scenarios where relevant datasets appear too low in the ranking. In dataset retrieval tasks, where each query can match multiple items, MAP provides a balanced measure of ranking effectiveness, ensuring that relevant datasets are retrieved and positioned effectively rather than focusing only on either early retrieval (Precision@K) or completeness (Recall@K).

4. Mean Reciprocal Rank (MRR) captures how quickly a user finds at least one relevant dataset. In many real-world scientific scenarios, discovering any suitable dataset early in the results list enables researchers to begin their work immediately. If the user's primary goal is to identify a single relevant dataset quickly, a higher MRR translates into a better user experience and less time spent filtering irrelevant results.

$$MRR = \frac{1}{|Q|} \sum_{q \in Q} \frac{1}{r_q} \tag{5}$$

where r_q is the rank position of the first relevant dataset for query q.

Our choice of evaluation metrics balances alignment with prior work, user expectations, and retrieval challenges specific to datasets. By using the same four core metrics Precision@K, Recall@K, MAP, MRR – as Viswanathan et al, we enable direct comparison with their baseline, allowing us to test whether our contributions improve retrieval effectiveness. Moreover, our choice of metrics captures both accuracy and comprehensiveness, ensuring our search engine not only retrieves the right datasets but also misses as few relevant documents as possible. Precision@K and MRR focus on how quickly and accurately a user finds a useful dataset, making them ideal for users searching for a single authoritative dataset; recall@K and MAP, on the other hand, evaluate how well the system retrieves all relevant datasets, which is critical for researchers who need multiple datasets for comparative analysis.

Following Viswanathan et al's methodology, the evaluation will be performed on the SciREX dataset [14], a set of 438 human-annotated papers and relevant datasets, converted into our 3-part document representation format detailed in subsection 3.2 *Document Creation Module*.

4 Experimental Design and Testing Plan

To refine the system, several experiments are conducted to test different approaches and determine optimal configurations. All of the configurations will be evaluated and compared based the 4 metrics specified in subsection *3.7 Evaluation Module*.

One major area of investigation is the preprocessing of queries and documents. The impact of query spelling correction will be assessed to determine whether it improves retrieval accuracy. Different ways of structuring dataset metadata will also be tested to find the most effective representation.

Another experiment will compare our weighted sum similarity scoring function with a simple similarity scoring function applied on document representations that merge the 3 components of our original representations (title, extended description, feature name summaries) into a single string. In this experiment, we introduce our first hypothesis: **the weighted sum similarity function has the highest performance across the board**. The rationale is that a single similarity function is the rough equivalent of a weighted sum similarity function where the weights are determined by the length of the text, and we see no reason why length of title, description, or feature name summary should correlate with relevance.

The performance of SPECTER2 will be evaluated against SciBERT to determine which model provides superior retrieval performance. Additionally, an experiment will assess whether fine-tuning SPECTER2 with the custom proximity adapter (as well as alternative adapter implementations) leads to better results compared to fine-tuning without any adapters. Since SPECTER2 was intended as an improvement on SciBERT, and is specifically optimised to work with adapters [15], we formulate our second hypothesis as follows: **SPECTER2 with the proximity adapter shows the highest performance across the board**.

Another key question involves the role of feature name summaries in retrieval performance. Tests will conducted to determine whether these summaries enhance the search engine's ability to retrieve relevant datasets or if they introduce unnecessary noise that degrades accuracy. Since feature name summaries provide additional information, allowing the search engine to retrieve more relevant documents, we formulate our third hypothesis as follows: **the weighted sum similarity function has the highest Recall@K**.

All the above experiments will be conducted on the validation set to mitigate data leakage and ensure reliable conclusions.

5 Novel Contributions

This search engine introduces several innovations that distinguish it from existing approaches. One of the major improvements is our enhanced document representation, which includes LLM-generated feature name summaries to provide additional context and retrieve relevant documents that are likely to have been missed by earlier approaches. Another key advancement is the expansion of the dataset by means of incorporating OpenML datasets and additional queries, ensuring greater diversity in the training data. We also SPECTER2 instead of SciBERT as our bi-encoder, which is expected to lead to more advanced document representations.

Finally, the traditional similarity-based ranking function is replaced with a weighted sum similarity function, allowing the search engine to balance different aspects of document relevance dynamically – which we expect to lead to improved performance.

With these architectural improvements, fine-tuning methodologies, and experimental evaluations, this search engine is designed to provide state-of-the-art dataset retrieval, leveraging the latest advancements in language modeling and ranking optimizstion.

6 Tools

We employ a Github repository to store all code and data collected, enabling collaboration between team members. For the writing of our report, we use Overleaf (an interface for LaTex), and for the collaboration on ideas and insights, we use Google Docs. For our computing environment, we use Google Colab to leverage its GPU capabilities. Depending on the time constraints, the backend will potentially be built using Flask, with CSS and HTML for frontend development. Python will be used to handle data processing, model integration, and retrieval logic. For dataset compilation, we will utilise the OpenML datasets API to fetch all the necessary dataset metadata – for the processing of which we will also require the JSON, NumPy, and Pandas libraries. The other datascets used in the training set generation and evaluation modules, respectively, are DataFinder Dataset [4] and SciREX

[14]. We will access the SPECTER2 bi-encoder via Hugging Face's Transformers library, using its adapter framework to create our custom proximity adapter [15]; we will also use the Transformers library to access Galactica. Model training and inference (in case of Galactica) are to be performed using PyTorch. For scoring functions, we will use rank-BM25 for the BM25 ranking function and scikit-learn for the evaluation metrics. Finally, for visualisation, we will rely on Matplotlib and Seaborn to present our results effectively.

7 Task Allocation and Timeline

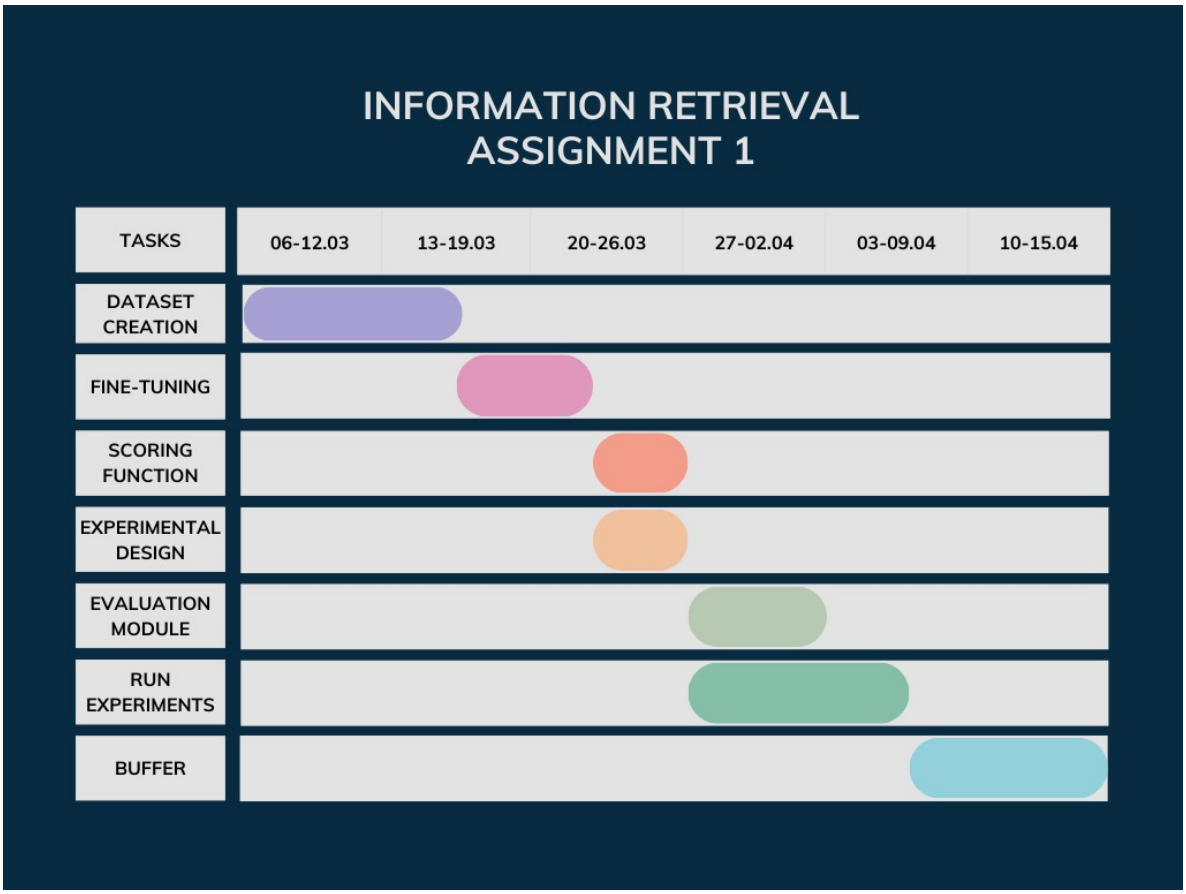

Figure 2: Project timeline

In the first week and a half (06-16/03), Angeline will implement the document creation module, and Maxim will work alongside her to create the query generation and training set generation modules. In the following week (17-23/03), Karl will work on fine-tuning SPECTER2, with Maxim assisting. The week after that (24-31/03), Maxim will implement the scoring function as described in subsection *3.2 Document Creation Module*. At the same, the entire team will gather to discuss the experimental design according to the current implementation, as there will likely be changes depending on the specifics of the implementation. Finally, Endro will integrate the evaluation module into the current system (27/03-02/04), while the rest of the team starts running experiments as per our experimental design (27/03-06/04). The ultimate 9-day buffer (07/04-15/04) is to finalise our report and correct for unforeseen delays.

References

[1] S. R. Kunze and S. Auer, "Dataset retrieval," in *2013 IEEE Seventh International Conference on Semantic Computing*, pp. 1–8, 2013.

[2] K. Sostek, D. M. Russell, N. Goyal, T. Alrashed, S. Dugall, and N. Noy, "Discovering datasets on the web scale: Challenges and recommendations for google dataset search," *Harvard Data Science Review*, Apr 2024. https://hdsr.mitpress.mit.edu/pub/psnc8zsr.

[3] M. Lu, S. Bangalore, G. Cormode, M. Hadjieleftheriou, and D. Srivastava, "A dataset search engine for the research document corpus," in *2012 IEEE 28th International Conference on Data Engineering*, pp. 1237–1240, 2012.

[4] V. Viswanathan, L. Gao, T. Wu, P. Liu, and G. Neubig, "DataFinder: Scientific dataset recommendation from natural language descriptions," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (A. Rogers, J. Boyd-Graber, and N. Okazaki, eds.), (Toronto, Canada), pp. 10288–10303, Association for Computational Linguistics, July 2023.

[5] J. Vanschoren, J. N. van Rijn, B. Bischl, and L. Torgo, "Openml: networked science in machine learning," *CoRR*, vol. abs/1407.7722, 2014.

[6] J. Keller and L. P. M. Munz, "Tekma at clef-2021: Bm-25 based rankings for scientific publication retrieval and data set recommendation," in *Conference and Labs of the Evaluation Forum*, 2021.

[7] D. Brickley, M. Burgess, and N. Noy, "Google dataset search: Building a search engine for datasets in an open web ecosystem," in *The World Wide Web Conference*, WWW '19, (New York, NY, USA), p. 1365–1375, Association for Computing Machinery, 2019.

[8] R. Taylor, M. Kardas, G. Cucurull, T. Scialom, A. Hartshorn, E. Saravia, A. Poulton, V. Kerkez, and R. Stojnic, "Galactica: A large language model for science," 2022.

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (J. Burstein, C. Doran, and T. Solorio, eds.), (Minneapolis, Minnesota), pp. 4171–4186, Association for Computational Linguistics, June 2019.

[10] A. Vaswani, N. M. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Neural Information Processing Systems*, 2017.

[11] I. Beltagy, A. Cohan, and K. Lo, "Scibert: Pretrained contextualized embeddings for scientific text," *CoRR*, vol. abs/1903.10676, 2019.

[12] A. Singh, M. D'Arcy, A. Cohan, D. Downey, and S. Feldman, "Scirepeval: A multi-format benchmark for scientific document representations," 2023.

[13] A. I. for Artificial Intelligence, "SPECTER2," 2021. `https://github.com/allenai/SPECTER2`.

[14] S. Jain, M. van Zuylen, H. Hajishirzi, and I. Beltagy, "SciREX: A challenge dataset for document-level information extraction," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, eds.), (Online), pp. 7506–7516, Association for Computational Linguistics, July 2020.

[15] Allen Institute for Artificial Intelligence, "Specter2." `https://huggingface.co/allenai/specter2`, 2023. Hugging Face Model Hub. Accessed: 2025-03-01.