Proceedings of the ASME 2022 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference
IDETC/CIE 2022
August 14–17, 2022

DETC2022/CIE-89557

SEMANTIC SEARCH WITH SENTENCE-BERT FOR DESIGN INFORMATION RETRIEVAL

Hannah S. Walsh
Intelligent Systems Division
NASA Ames Research Center
Moffett Field, California

Sequoia R. Andrade
HX5, LLC.
Moffett Field, California

ABSTRACT

Managing and referencing design knowledge is a critical activity in the design process. However, reliably retrieving useful knowledge can be a frustrating experience for users of knowledge management systems due to inherent limitations of standard keyword-based searches. In this research, we consider the task of retrieving relevant lessons learned from the NASA Lessons Learned Information System (LLIS). To this end, we apply a state-of-the-art natural language processing (NLP) technique for information retrieval (IR): semantic search with sentence-BERT, which is a modification of a Bidirectional Encoder Representations from Transformers (BERT) model that uses siamese and triplet network architectures to obtain semantically meaningful sentence embeddings. While the pre-trained sBERT model performs well out-of-the-box, we further fine-tune the model on data from the LLIS so that it learns on design engineering-relevant vocabulary. We quantify the improvement in query results using both standard sBERT and fine-tuned sBERT over a keyword search. Our use case throughout the paper is to use queries related to specific requirements from a NASA project. Fine tuning the sBERT model on LLIS data yields a mean average precision (MAP) of 0.807 *on queries based on information needs from a real NASA project. Results indicate that applying state-of-the-art natural language processing techniques, especially when fine-tuned using engineering data, to design information retrieval tasks shows significant promise in modernizing design knowledge management systems.*

1 INTRODUCTION

Leveraging repositories of design information is critical to informing new designs. Knowledge management assists an organization in fully utilizing its intellectual assets. NASA's Lessons Learned Information System (LLIS) [1], one of NASA's knowledge management systems, contains thousands of lessons learned across all NASA centers and is a valuable resource for project managers and engineers. However, past reviews of NASA's LLIS indicate that the system is underutilized, in part because it is "not user friendly" and is "unhelpful" [2]. While there are a variety of reasons for these judgments, one method for improving the usability and usefulness of such systems is by improving its information retrieval (IR) capabilities. Particularly with large repositories, it can be difficult for users to retrieve relevant information using standard keyword or metadata searches. Consequently, the usefulness of such repositories is diminished if users are unable to efficiently search for the information they need.

Recent advances in natural language processing (NLP) have enabled significant improvements in information retrieval capabilities. Bi-directional Encoder Representations from Transformers (BERT) [3] models have in recent years revolutionized NLP and show promise in improving current knowledge management systems for design. BERT models are trained using a Masked Language Model (MLM) in which the models learn to predict masked words, thereby learning context from both left-to-right and right-to-left (i.e., a "bi-directional" encoder). BERT models are trained over thousands or millions of documents from sources covering diverse domains such as Wikipedia so that they are broadly applicable. Semantic search with sentence embeddings using siamese BERT networks (sentence-BERT, or sBERT) [4] is a BERT-based method suited to the task of information retrieval.

However, the specialized technical vocabulary common in engineering documentation hinders the usefulness of information retrieval systems that have not been specifically trained on or do not contain ontologies for engineering concepts. Consequently, a pre-trained sBERT model has likely not seen specialized engineering vocabulary in context, which could adversely affect performance for engineering applications. While it is prohibitively expensive to train an sBERT model from scratch, fine-tuning allows sBERT models to be specialized for a particular task, such as matching job candidates with positions [5]. One of the advantages of BERT models is that fine-tuning does not require substantial re-architecting of the model; instead, only one additional output layer is required [3]. Moreover, while training sBERT models from the ground up is too expensive for most individual tasks, fine-tuning requires only a fraction of the data and resources.

In this research, we present a semantic search capability enabled by sBERT that has been fine-tuned using documents from the NASA LLIS, which contains publicly available lessons learned from NASA. This model utilizes the state-of-the-art in natural language processing, while being specifically tuned to the needs of design information retrieval. While this model is tuned for LLIS searches, it can be applied to other repositories with content similar to the LLIS; alternatively, the model may be further fine-tuned (building on the model fine-tuned using LLIS data) on additional design engineering specialized data. We show and quantify the improvement of the fine-tuning compared to the baseline, pre-trained sBERT model (not fine-tuned on LLIS data).

2 BACKGROUND

Information retrieval in design is the task of obtaining resources relevant to an information need from a repository of such resources. There are variably sophisticated ways in which information can be retrieved, including indexing [6] and hierarchical thesauri [7]. Ontology-based information retrieval is also popular [8]. In engineering design, much emphasis has been placed on semantic networks, which represent knowledge and links between information as a collection of nodes and edges [9]. Semantic networks have been used in the past by NASA engineers to create SemanticOrganizer, a web application for knowledge management, search, and document organization across a range of projects and users [10]. Further development created XSearch, a platform that allows users (i.e., flight controllers) to search multiple mission-relevant databases with a signal query [11]. This process relies on textual similarity in addition to semantic networks, and also recommends cross-referenced documents via outbound and inbound citation identification. These methods may outperform keyword searches [12], but are dependent on the construction and continued relevance of the networks. Network analysis metrics can be applied to understand knowledge relations contained in these forms [13]. Some types of networks can be generated dynamically [14] or can be learned from engineering design-specific data [15]. The networks that are specialized for engineering-specific applications tend to perform best.

While some intelligent search capabilities have been proposed [16], information retrieval in design has not kept pace with the state-of-the-art in artificial intelligence. Natural language processing in particular has seen significant strides in recent years largely due to the use of BERT models. As much of the documentation relevant to engineering design is stored, at least in part, in natural language format, the application of BERT models to information retrieval in design shows promise. BERT models are trained on enormous amounts of diverse data, making them more powerful than most models built from scratch for a particular task. BERT and its variations have been applied to several different tasks within natural language processing, including information retrieval [17–19]. sBERT is a more recent adaptation that utilizes a siamese (i.e., parallel processing of two or more texts) BERT-network to obtain sentence embeddings that are semantically meaningful, a variation that is particularly well-suited to semantic search [4]. Applied to semantic similarity tasks, sBERT demonstrates significant computational gains over standard BERT [4]. sBERT-enabled semantic search has been successfully applied to specialized queries, including coronavirus information retrieval [20] and information retrieval for infrastructure damage queries [21]. Semantic search differs fundamentally from keyword searches in that it considers meaning and context instead of only exact matches of a word. Equivalently, semantic search is more comparable to how a human would find links from query to search result.

3 METHODOLOGY

This paper uses semantic search with fine-tuned sBERT to retrieve relevant lessons from the NASA LLIS. Unless otherwise noted, all steps are implemented using sentence-transformers in Python [4]. Information retrieval can be defined as a process by which a system obtains documents relevant to a user's query. In design, the documents a user may want to search take many forms. In this work, we consider documented lessons learned in NASA's LLIS. Design case studies and notebooks may also be useful to query [7]. Searches can use the text of the documents and/or metadata associated with those documents. In the case of the LLIS, there is limited metadata, but it is not available for all documents in the data set. Information retrieval is valuable for large sets of documents in which manually searching documents is inconvenient or untenable. An effective information retrieval system can improve the value of the stored data by increasing its accessibility to users of that data. The methodology is summarized in Fig. 1.

3.1 Use Case and Query Formulation

We consider a use case for this search as follows: a project manager on a NASA project needs to capture relevant lessons learned from past projects to identify risks and possible mitigation strategies in a new project. We adopt this use case for the scope of this paper; however, the information retrieval task is useful throughout the design process and a system's life cycle. The information needs considered in this paper are based on real requirements documented for NASA's Human Landing System program [22], specifically the first three listed in the functional and performance requirements. The project manager is searching for project risks relevant to the following general areas:

1. ***Cyber security***: This information need is specifically related to cyber security, which includes information technology, operational technology, and other systems specialized for HLS [22]. It does not include physical security such as building security or security personnel at launch sites.
2. ***Fault tolerance***: This information need covers strategies for fault tolerance, including hazard and risk analysis needed to decide the level of failure tolerance required as well as specific implementation methods [22].
3. ***Fault isolation***: This information need relates to isolation and recovery from faults [22], which is distinct from *fault tolerance*, above. We include these two closely related information needs to test how the information retrieval system distinguishes between queries with similar words, but different concepts.

An effective query is a representation of a user's information need. In practice, not all queries are effective representations of this information need. While in-depth strategies for query formulation are outside the scope of this paper, we conducted small trial-and-error studies to ensure our chosen queries are good representations of the information needs outlined in the HLS requirements. These trials led to the decision to augment the basic topic of the query with additional keywords to further refine the information need. For instance, we are looking for methods to improve the fault tolerance of the system rather than definitions of fault tolerance. We adopt the following three queries based on the information needs listed above:

1. "Cyber security data and systems"
2. "Fault tolerance methods"
3. "Fault isolation methods"

3.2 Search Formulation

A key decision point in formulating the problem is whether the search is symmetric or asymmetric. A symmetric search involves a query and search result of roughly equivalent length; a rule of thumb to determine whether a search is symmetric is to ask whether the query and result could be reversed and for the problem to still be logical. An example of a symmetric search is retrieving similar scientific articles to a queried article. An asymmetric search, in contrast, involves querying the system with a short phrase and retrieving longer documents. In this research, we consider the latter. While it may be useful to input longer documents, such as requirements, to the search system, this places restrictions on what is required of the user in order to search the system and, therefore, at which phase of the design process the search can be conducted. Instead, by allowing a flexibly formatted, short phrase as the query, our search system can be used flexibly and, especially, early in the design process, before substantial project documentation is available. A final consideration is that, since the encoder limits the maximum length of the documents to 512 tokens, we search passages rather than entire documents that are relevant to the query. Traceability is maintained such that, when a relevant passage is found, the entire document can be retrieved, with the relevant passage highlighted. For this problem, we define a passage as a subsection of the lessons learned document. All unstructured, text-based (i.e., non-metadata) subsections of the documents are considered, at least for those lessons for which they are completed. These are: Abstract, Driving Event, Lesson(s) Learned, Recommendation(s), and Evidence of Recurrence Control Effectiveness.

3.3 sBERT Model Selection

There are options for pre-trained sBERT models to use in the search system hosted on Huggingface. In general, several factors must be considered when selecting a pre-trained model. First is whether the model is trained for symmetric or asymmetric search. The inputs planed for use with the model should be in the same format as those used to train the model. Multi-QA models generalize well because they are trained on extensive (215M) question-answer pairs from multiple sources, including Stack Exchange, Quora, WikiAnswers, and Yahoo Answers. MSMARCO models are built from a large information retrieval corpus created from real user queries and also tend to generalize well. Models may be tuned either for cosine similarity, which measures the cosine of the angle between two vectorized documents (D_i, D_j) in Equation 1, or dot product. Models tuned for cosine similarity tend to prefer the retrieval of shorter documents, while dot product models prefer longer documents. Other models use normalized vectors which can be used with either cosine similarity or dot product. Finally, a model will need to be selected that is with appropriate language(s) for the application. Our application includes only English-language documents, so we do not opt for a multi-lingual model. With these considerations, we select the Multi-QA model multi-qa-mpnet-base-cos-v1, which has slightly higher performance compared to similar

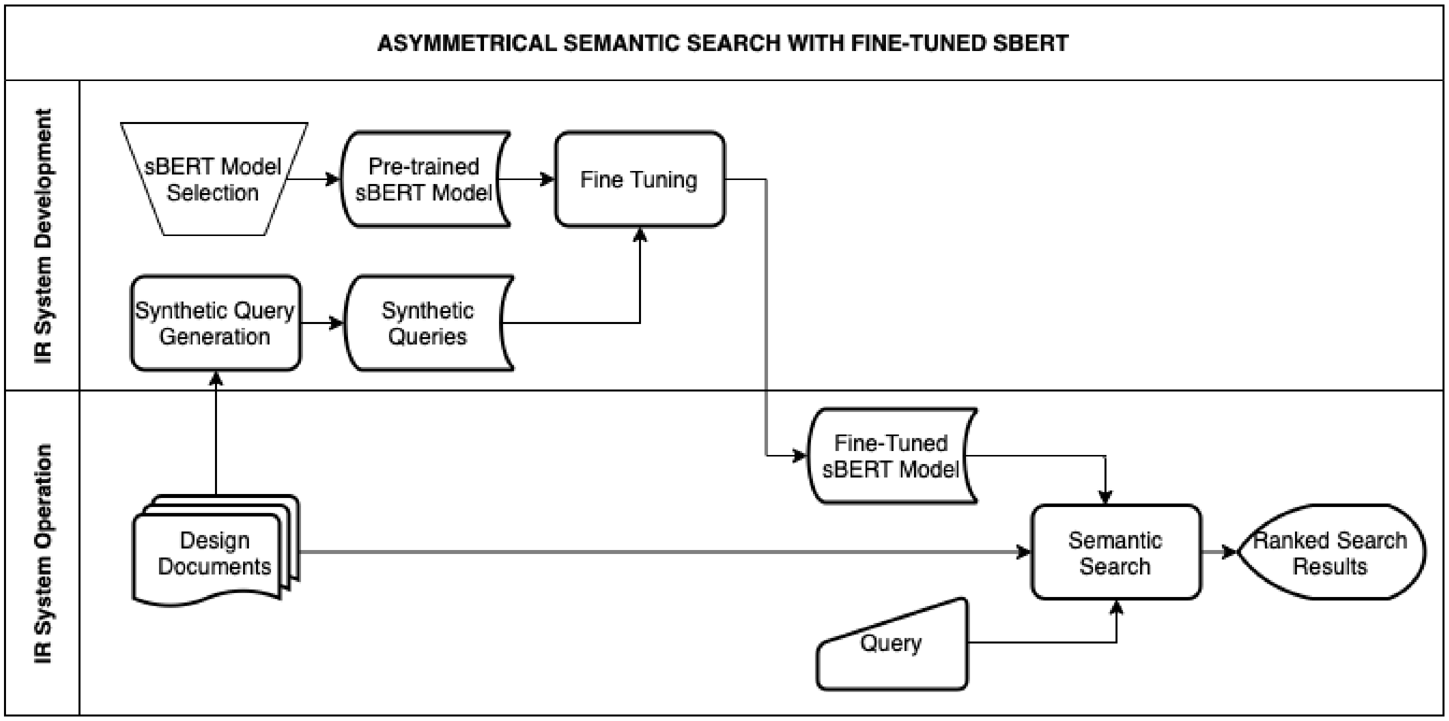

FIGURE 1: Summary of the methodology. Methodology is shown generally for asymmetric search tasks. In this research, design documents are lessons learned from the NASA LLIS [1].

models at a marginal cost to speed. This model additionally is appropriate for dot-product, cosine similarity, or Euclidean distance measures, providing a degree of flexibility in our query system.

$$cos\theta = \frac{D_i\dot{D}_j}{||D_i||||D_j||} \tag{1}$$

3.4 Fine-Tuning

The selected base sBERT model is fine-tuned on the LLIS. Fine-tuning models meant for asymmetric search requires pairs of queries and results. In many cases, as in ours, this data is unavailable. It would require a prohibitive amount of time and effort for humans to generate queries for the LLIS documents. Instead, we use a method for synthetic query generation, which creates a synthetic query given a document, thereby allowing us to fine-tune the model when training data is unavailable [23]. The queries are generated using docTTTTTquery [24], an improvement to doc2query that uses T5 [25] as the expansion model. docTTTTTquery has been trained on an MS MARCO [26] model from BeIR, specifically query-gen-msmarco-t5-large-v1 [27]. A small subset of synthetically generated queries is given in Table 1. Three queries are produced per document (in our case, passage since we decided to subdivide the documents). This gives 31,530 total pairs for training, with a small subset withheld for the evaluator.

Some models require annotations for each training pair. For instance, a similarity score may be provided. In the case of information retrieval, no such annotations are required. Instead, the loss function Multiple Negatives Ranking Loss, which is suitable for information retrieval tasks, takes as input pairs of queries and results. For each query, it assumes that all results with which it is paired are positive pairs and that those that are not specified as pairs are negative pairs. An Information Retrieval Evaluator is used to evaluate the model during training. This evaluator retrieves the top k most similar documents to each query by measuring Mean Reciprocal Rank, Recall@k, and Normalized Discounted Cumulative Gain (NDCG). The model is fine-tuned for two epochs and the evaluator is run every one-hundred iterations. The fine-tuned model can be saved and referenced for semantic search in the same manner as the pre-trained model.

Fine-tuning, unlike training a BERT model from scratch, is attainable given moderate computing power. Moderate GPU usage can accelerate the process considerably. Depending on your computing power, it may be necessary to decrease the batch size, which increases run time. It is usually recommended to train using larger batches where possible. In our experience fine-tuning for this paper, switching from CPU to moderate GPU dropped the run time from being on the order of days to under an hour.

TABLE 1: Subset of synthetically generated queries, replicated exactly from the algorithm output other than adding capitalization to acronyms for readability.

Synthetic Query	Lesson Excerpt
What can you test for HPH	Trace contaminants in high-purity hydrazine (HPH) propellant impact a wide variety of commercial, Department of Defense (DoD), and NASA missions. Depending on thruster design, contaminants must be kept at extremely low levels and are verified as such by routine analysis...
What would happen if the propulsion subsystem fail	Propulsion subsystem check valves on the Juno spacecraft malfunctioned during preparations for a bi-propellant main engine orbital maneuver. Although the failure mechanism had no major impact on the Juno mission, it poses a risk that an engine may operate outside of its qualified mixture ratio, which could lead to mission loss...
Why did my VFM go wrong during welding	A failure occurred during the first attempt at welding of the Europa Clipper Venturi Flow Meter (VFM) flight units. During the first pass, excessive heat input to the welding area caused the weld root reinforcement material to melt. This left a divot on the top surface and an obstruction in the internal flow passage of the VFM...

We use a batch size of 32. A smaller batch size of 16 was also tested, but this increased the fine tuning time (38 minutes), while a larger batch size of 64 required more memory than we had available.

3.5 Semantic Search

Once the model is fine-tuned and saved, it can be easily called for semantic search. The model is used to generate embeddings for the corpus. These embeddings can be stored and referenced for each search – while testing our model, we found the corpus embeddings to take a majority of the total time taken to perform a semantic search (not including model fine-tuning steps) before we started saving them for future runs, as shown in Table 2. Embeddings for the query are obtained in the same manner. Once all embeddings are obtained, a cosine similarity search is performed between the list of query embeddings and corpus embeddings, as depicted in Figure 2. Cosine similarity search is effective for corpora up to about one million entries [4]. The top *k* hits may be returned (ranking all results is possible, but computationally inefficient and of diminishing utility).

TABLE 2: Summary of computational time for each step in the methodology. Synthetic query generation, model fine tuning, and corpus embeddings can and should be stored and reused. The query embedding and semantic search must run for each new query to the system.

Step	Time
Synthetic query generation	156 minutes
sBERT model fine tuning	31 minutes
Corpus embeddings	30 seconds
Query embedding	0.086 seconds
Semantic search	0.014 seconds

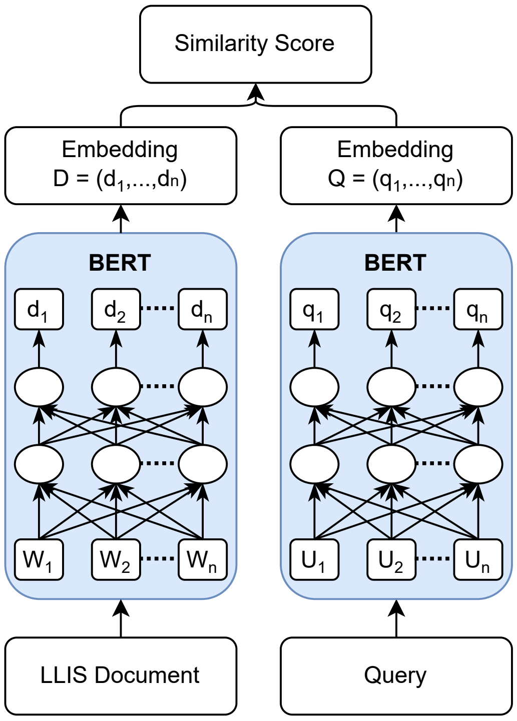

FIGURE 2: sBERT model architecture for semantic search. The siamese model applies BERT to both the query and LLIS document to produce encoded text in a vector, then their similarity is calculated using the chosen metric. For a given query, this is performed for each LLIS document to rank results.

4 RESULTS

Query results for the first query, “cyber security data and systems”, using both the pre-trained (not fine-tuned) and fine-tuned sBERT models are given in Table 3. Note that cosine similarity scores (“Score” in Table 3) for ranked results are calculated using the respective models. Table 3 provides the top three ranked results for each model. These results share only one lesson in common; however, all top three results from both models are judged as relevant by a human reader.

Analysis uses basic information retrieval definitions of precision and recall with some specializations since the IR system retrieves ranked results (rather than an unordered set) and the dataset is large. Precision P is defined in an IR context in Eq. 2 [7]. In Eq. 2, RET is defined as the set of documents retrieved by the information retrieval system for a given query and REL is the set of documents that are relevant to that query. REL is typically (as well as in this paper) determined among human experts [7]. A precision-recall curve (recall defined in Eq. 3) over $0 < k < 30$ for the three queries is given in Fig. 3. Jagged sections of the graph indicate that as k is increased, a new, relevant document is found, increasing both the recall and precision. If an irrelevant document is instead found, recall is constant but precision decreases. Figure 3 indicates that the fine-tuned model has higher precision compared to the pre-trained model at the same recall levels for all three queries tested.

$$P = \frac{RET \cap REL}{RET} \tag{2}$$

$$R = \frac{RET \cap REL}{REL} \tag{3}$$

We provide precision at $k = 10$, $k = 20$, and $k = 30$ documents for the three queries for each information retrieval system in Table 4. The rationale behind the precision at k metric is that the user of an information retrieval system will desire relevant results within the first page or two (i.e., within relatively few total results). Precision at k is a meaningful measure of this need because it quantifies how relevant the first few results will be. The fine-tuned sBERT model performs better than the pre-trained sBERT model at each $k = 10$, $k = 20$, and $k = 30$ for all three queries, with a perfect (1.000) precision for the first ten results for the first and second queries. We note the larger gap between the fine-tuned and baseline performance in the third query compared to the second query. The third query is the most challenging information retrieval task, as fault isolation is similar in context and semantics but distinct in concept to fault tolerance. Moreover, our sampling suggests there are fewer overall lessons relevant to fault isolation than to fault tolerance. It is likely that fine-tuning is especially useful for distinguishing between these two concepts (fault tolerance and isolation) given their specific engineering meanings.

Mean average precision (MAP) is given in Eq. 4, where n_q is the number of queries tested, $n_{d,j}$ is the number of relevant documents for a query j, and P_i is the precision at i. For MAP, unlike precision at k, the entire set of relevant documents is required so that precision can be averaged across all recall levels. The authors manually searched the data set for relevant lessons. For the first query, there are forty-five total relevant documents, for the second, one-hundred forty relevant documents, and for the third, thirty-five relevant documents. MAP for each model is given in Table 5. Fine-tuning outperforms the baseline pre-trained model at a MAP of 0.807 compared to 0.648 across the queries tested.

$$MAP = \frac{1}{n_q} \sum_{j=1}^{n_q} \frac{1}{n_{d,j}} \sum_{i=1}^{n_{d_j}} P_i \tag{4}$$

5 DISCUSSION

Although we find both the pre-trained and fine-tuned models perform well for the given task, the fine-tuned model in particular shows superior performance. Given the relative ease of fine-tuning the model given moderate GPU, the results indicate that fine-tuning is worthwhile. In particular, the precision of the fine-tuned model for low k is especially high and indicates that users will be able to use the system to find relevant results quickly (i.e., without scrolling through many irrelevant results). One implementation challenge is that sBERT-enabled semantic search methods have not been integrated with existing search systems for individual repositories, such as the LLIS. Existing repositories may store documents in non-Python database formats, such as SQL or cloud-based services, and thus additional architecture may be necessary to query these databases using Python-based sBERT semantic search.

6 CONCLUSIONS AND FUTURE WORK

This work has presented a query system for design information retrieval using sentence transformers. We fine-tune the model for information retrieval from NASA’s Lessons Learned Information System, which improves performance compared to the baseline pre-trained model when tested using precision vs. recall curves, precision at k, and MAP. Our results indicate that the fine-tuned sBERT model is a viable solution to the stated IR problem. The application of the state-of-the-art in natural language processing to design information retrieval will improve efficiency of design information use and re-use.

While this work focuses on applying bi-directional encoders to information retrieval, they can also be applied to knowledge

TABLE 3: Query results for the first query, "cyber security data and systems", from the pre-trained model and fine-tuned model. Though the two models only share one top-three result, all top-three results from both models belong to *REL* (i.e., are judged relevant to the query by a human reader).

Query 1: Cyber security measures for data and systems				
Pre-Trained Model				
Rank	**Score**	**Lesson**	**Lesson Title**	**Passage Excerpt**
1	0.542	1150	Agency-Wide/ Computer Hardware-Software/ Computer Security	NASA concurs in principle with both parts of this recommendation. Regarding analysis with the National Security Agency (NSA)...
2	0.523	1149	Agency-Wide/ Computer Hardware-Software/ Computer Security	Implementation of NASA Agency-Wide Computer Security Plan...
3	0.517	1250	Network Security/ Reduction of Vulnerabilities/ Penetration Exercises	The terrorist attacks on September 11 emphasized the need for increased security of all national assets including NASA's computer systems...
Fine-Tuned Model				
Rank	**Score**	**Lesson**	**Lesson Title**	**Passage Excerpt**
1	0.517	1250	Network Security/ Reduction of Vulnerabilities/ Penetration Exercises	The terrorist attacks on September 11 emphasized the need for increased security of all national assets including NASA's computer systems...
2	0.513	1175	Computer Hardware-Software/ System Security/Personnel Awareness and Training	16a. Complete and maintain security plans for all appropriate computer systems and ensure that the computer security program is sustaining...
3	0.469	1250	Network Security/Reduction of Vulnerabilities/Penetration Exercises	Accelerate the schedule of penetration exercises to gain greater insights into computer security vulnerabilities...

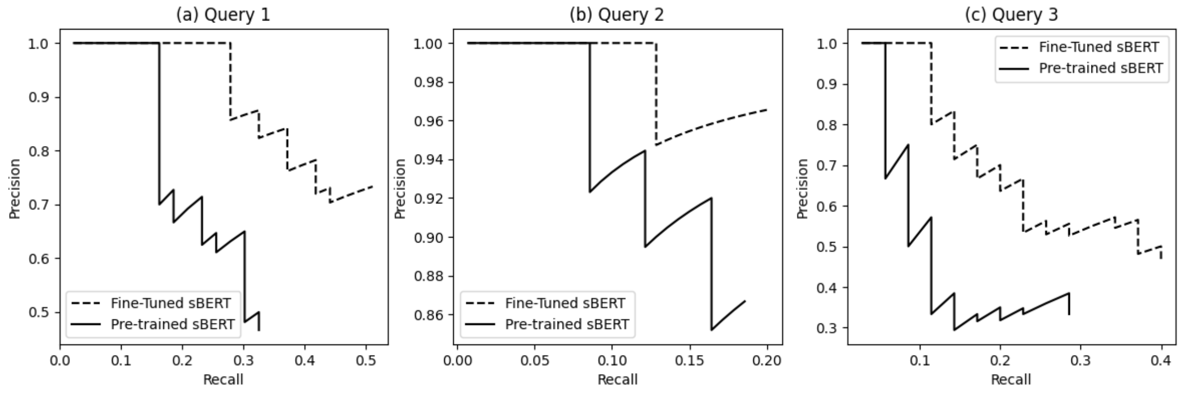

FIGURE 3: Precision versus recall plotted for $1 < k < 30$ for fine-tuned and pre-trained sBERT models. The fine-tuned model offers higher precision at equivalent recall values compared to the pre-trained model.

discovery. For instance, BERTopic can be applied to discovering topics within the LLIS and other repositories of design-related information, as has previously been done with more conventional natural language processing approaches such as Latent Dirichlet Allocation (LDA) [28]. Additionally, while a query system is undoubtedly essential to a knowledge management system, a *proactive* knowledge delivery system will more effectively exploit the knowledge available to designers. This would mean

TABLE 4: Precision at *k* for the first three queries. Fine-tuned sBERT outperforms the baseline model on all three queries and at all *k* levels.

Query 1: cyber security of data and systems			
	Precision at:		
IR Method	$k = 10$	$k = 20$	$k = 30$
Pre-trained sBERT	0.700	0.650	0.466
Fine-tuned sBERT	1.000	0.800	0.733

Query 2: fault tolerance methods			
	Precision at:		
IR Method	$k = 10$	$k = 20$	$k = 30$
Pre-trained sBERT	1.000	0.900	0.866
Fine-tuned sBERT	1.000	0.950	0.965

Query 3: fault isolation methods			
	Precision at:		
IR Method	$k = 10$	$k = 20$	$k = 30$
Pre-trained sBERT	0.400	0.350	0.333
Fine-tuned sBERT	0.700	0.550	0.466

TABLE 5: Mean average precision (*MAP*) for each IR method. Fine-tuned sBERT outperforms the baseline model.

IR Method	**MAP**
Pre-trained sBERT	0.648
Fine-tuned sBERT	0.807

anticipating users' needs based on design characteristics, scope, or other attributes. One possibility is matching current design documentation to lessons learned using a symmetric search formulation. These capabilities will be addressed in the broader intelligent knowledge management toolkit the team is developing.

ACKNOWLEDGMENT

This research is supported by the System-Wide Safety (SWS) project in the Airspace Operations & Safety program (AOSP) in the NASA Aeronautics Research & Mission Directorate (ARMD). Any opinions or findings of this work are the responsibility of the authors and do not necessarily reflect the views of the sponsors or collaborators. The authors would like to thank Dr. Guillaume Brat for his input.

REFERENCES

[1] NASA Lessons Learned Information System (LLIS). `https://llis.nasa.gov`. Accessed: 2022-02-09.

[2] Office of Inspector General, 2012. Review of NASA's Lessons Learned Information System. Tech. Rep. IG-12-012, Office of Audits.

[3] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K., 2019. BERT: Pre-training of deep bidirectional transformers for language understanding.

[4] Reimers, N., and Gurevych, I., 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks.

[5] Lavi, D., Medentsiy, V., and Graus, D., 2021. "conSultantBERT: Fine-tuned Siamese sentence-BERT for matching jobs and job seekers". *ArXiv,* ***abs/2109.06501***.

[6] Ruecker, L., and Seering, W. P., 1992. "User-Oriented Intelligent Indexing and Retrieval in a Design Documentation System". Vol. ASME 1992 6th Annual Database Symposium: Engineering Data Management — Key to Integrated Product Development of *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pp. 21–28.

[7] Wood, W. H., Yang, M. C., Cutkosky, M. R., and Agogino, A. M., 1998. "Design Information Retrieval: Improving Access to the Informal Side of Design". Vol. Volume 3: 10th International Conference on Design Theory and Methodology of *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. V003T03A001.

[8] Li, L., Qin, F., Gao, S., and Qin, X., 2014. "Ontology-Based Design Rationale Retrieval Supporting Natural Language Query". Vol. Volume 1B: 34th Computers and Information in Engineering Conference of *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. V01BT02A017.

[9] Han, J., Sarica, S., Shi, F., and Luo, J., 2020. Semantic networks for engineering design: A survey.

[10] Keller, R., Berrios, D., Carvalho, R., Hall, D., Rich, S., Sturken, I., Swanson, K., and Wolfe, S., 2004. "SemanticOrganizer: A customizable semantic repository for distributed NASA project teams". In The Semantic Web - ISWC 2004: Third International Semantic Web Conference, Vol. 3298, pp. 767–781.

[11] Keller, R., Wolfe, S., Windrem, M., and Berrios, D., 2008. "XSearch: A system for searching and interrelating nasa mission operations data". In SpaceOps 2008 Conference.

[12] Li, Z., Raskin, V., and Ramani, K., 2008. "Developing engineering ontology for information retrieval". *Journal of Computing and Information Science in Engineering,* ***8***, March.

[13] Shi, F., Chen, L., Han, J., and Childs, P., 2017. "Implicit Knowledge Discovery in Design Semantic Network by Applying Pythagorean Means on Shortest Path Search-

ing". Vol. Volume 1: 37th Computers and Information in Engineering Conference of *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. V001T02A053.

[14] Ahmed, S., 2006. "An Approach to Assist Designers With Their Queries and Designs". Vol. Volume 4a: 18th International Conference on Design Theory and Methodology of *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pp. 359–366.

[15] Shi, F., Chen, L., Han, J., and Childs, P., 2017. "A Data-Driven Text Mining and Semantic Network Analysis for Design Information Retrieval". *Journal of Mechanical Design,* ***139***(11), 10. 111402.

[16] Nagasawa, S., Fukuzawa, Y., Miyata, Y., Hasegawa, H., and Sakuta, H., 1999. "Associative Transformation of Query Words for Case Retrieval System Based on Neural Network". Vol. Volume 2: 19th Computers and Information in Engineering Conference of *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pp. 483–489.

[17] Akkalyoncu Yilmaz, Z., Wang, S., Yang, W., Zhang, H., and Lin, J., 2019. "Applying BERT to document retrieval with Birch". In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations, Association for Computational Linguistics, pp. 19–24.

[18] Yang, W., Zhang, H., and Lin, J., 2019. Simple applications of BERT for ad hoc document retrieval.

[19] Westermann, H., Savelka, J., and Benyekhlef, K., 2021. "Paragraph similarity scoring and fine-tuned BERT for legal information retrieval and entailment". In New Frontiers in Artificial Intelligence, N. Okazaki, K. Yada, K. Satoh, and K. Mineshima, eds., Springer International Publishing, pp. 269–285.

[20] Esteva, A., Kale, A., Paulus, R., Hashimoto, K., Yin, W., Radev, D., and Socher, R., 2021. "COVID-19 information retrieval with deep-learning based semantic search, question answering, and abstractive summarization". *NPJ digital medicine,* ***4***(1), pp. 1–9.

[21] Kim, Y., Bang, S., Sohn, J., and Kim, H., 2022. "Question answering method for infrastructure damage information retrieval from textual data using bidirectional encoder representations from transformers". *Automation in Construction,* ***134***, p. 104061.

[22] National Aeronautics and Space Administration, 2021. NextSTEP-2 Appendix N: Sustainable Human Landing System Studies and Risk Reduction. `https://sam.gov/opp/a69b323be5494af2b04b85a1f5111076/view`. Online; accessed 28 January 2022.

[23] Thakur, N., Reimers, N., Rücklé, A., Srivastava, A., and Gurevych, I., 2021. BEIR: A heterogenous benchmark for zero-shot evaluation of information retrieval models.

[24] Nogueira, R., Lin, J., and Epistemic, A., 2019. "From doc2query to docTTTTTquery". *Online preprint*.

[25] 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.

[26] Reimers, N., and Gurevych, I., 2021. "The curse of dense low-dimensional information retrieval for large index sizes". In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), Association for Computational Linguistics, pp. 605–611.

[27] Thakur, N., Reimers, N., Rücklé, A., Srivastava, A., and Gurevych, I., 2021. "Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models". In Thirty-fifth Conference on Neural Information Processing Systems (NeurIPS 2021) - Datasets and Benchmarks Track (Round 2).

[28] Andrade, S. R., and Walsh, H. S., 2021. "Knowledge Discovery for Early Failure Assessment of Complex Engineered Systems Using Natural Language Processing". Vol. Volume 2: 41st Computers and Information in Engineering Conference (CIE) of *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. V002T02A061.