scientific reports

Check for updates

OPEN

Benchmarking pre-trained text embedding models in aligning built asset information

Mehrzad Shahinmoghadam✉ & Ali Motamedi✉

Accurate mapping of the built asset information to various data classification systems and taxonomies is crucial for effective asset management, whether for compliance at project handover or ad-hoc data integration scenarios. Due to the complex nature of built asset data, which predominantly comprises technical text elements, this process remains largely manual and reliant on domain expert input. Recent breakthroughs in contextual text representation learning (text embedding), particularly through pre-trained large language models, offer promising approaches that can facilitate the automation of cross-mapping of the built asset data. However, no comprehensive evaluation has yet been conducted to assess these models' ability to effectively represent the complex semantics specific to built asset technical terminology. This study presents a comparative benchmark of state-of-the-art text embedding models to evaluate their effectiveness in aligning built asset information with domain-specific technical concepts. Our proposed datasets are derived from two renowned built asset data classification dictionaries. The results of our benchmarking across six proposed datasets, covering clustering, retrieval, and reranking tasks, showed performance variations among models, deviating from the common trend of larger models achieving higher scores. Our results underscore the importance of domain-specific evaluations and future research into domain adaptation techniques, with instruction-tuning as a promising direction. The benchmarking resources are published as an open-source library, which will be maintained and extended to support future evaluations in this field.

Keywords Text embedding, Large language models (LLMs), Representation, Pre-trained language model, Information alignment, Domain-specific evaluation, Dataset, Benchmark

Asset management plays a pivotal role in ensuring optimal performance and extended life span of the built environment through a systematic process of monitoring and maintaining various facilities and equipment. The rapid advancement of digital technologies has led asset owners to increasingly demand enriched digital twins at project handover to support real-time operations and maintenance of the built assets[1]. Simultaneously, the growing awareness of the benefits of digitized asset management highlights the essential need for federated access to built asset data[2]. This requires aligning extensive data sources and their underlying schema with established data models, classification systems, or taxonomies to facilitate data accessibility for diverse stakeholders and improve interoperability across various software environments. In the context of the present study, "alignment" specifically refers to the task of accurately associating textual descriptions of built asset entities (e.g., building components, materials, equipment) with corresponding concepts or classes within an established target classification system. Such alignment involves capturing semantic equivalences or near-equivalences between descriptions across different terminologies and taxonomies, ensuring that textual data can be reliably mapped to ad-hoc or standardized domain-specific categories. However, aligning built asset data with pre-defined classification systems poses significant challenges in practice. A key challenge stems from the multi-source and multi-disciplinary nature of built asset data, which leads to the use of diverse formats and terminologies across different projects and stakeholders. For example, the terminology that architects utilize to describe the specifications for a particular building component or system can vastly differ from those used by structural engineers or subcontractors. Moreover, the structures of domain-specific classifications used in different disciplines often vary in granularity. For instance, the detailed engineering descriptions of an HVAC (Heating, Ventilation, and Air Conditioning) system provided by mechanical engineers may be far more comprehensive than those required and used by operations and maintenance teams. Finally, variations in local regulations and standards can further complicate the alignment process, particularly for large-scale or international projects.

Department of Construction Engineering, École de Technologie Supérieure, Montreal H3C 1K3, Canada. ✉email: mehrzad.shahinmoghadam.1@ens.etsmtl.ca; ali.motamedi@etsmtl.ca

These issues, combined with the dynamic and evolving nature of built asset data throughout an asset's lifecycle, lead to potential inconsistencies when integrating this data into a unified digital asset management environment.

In response, there have been several initiatives aimed at facilitating the digital delivery of built asset information while ensuring its conformity with predefined or standardized descriptions (data models, taxonomies, etc.). One major initiative is buildingSMART Data Dictionary (bSDD)[3], an international and ongoing effort whose main objective is to create shared definitions for describing the built environment. This is achieved through a collection of interconnected data dictionaries that are both human-readable and machine-readable[3]. Although making various data dictionaries programmatically accessible will facilitate access to agreed and consistent terms, the complexity and dynamic diversity of the built asset terminology necessitate robust data mapping strategies to accommodate various data descriptions and updates[4]. As a result, the asset information alignment process remains predominantly manual, relying heavily on the expertise of domain specialists to accurately map complex technical data[5]. The significant challenges associated with the manual alignment process, including high costs, time consumption, and potential for human error, highlight the need for more automated and reliable data mapping solutions.

The central thesis of our research builds upon the argument that recent advancements in natural language processing/understanding research can significantly enhance automated data mapping processes. In particular, the rich and contextualized representation of textual inputs as numeric vectors, commonly known as text embedding[6,7], provides advanced capabilities for machines to understand the semantics of the intricate terminologies. The numerical encoding of language understanding can then be leveraged to support various applications such as information retrieval, semantic similarity assessment, or clustering of textual data[8]. The enhanced capabilities of state-of-the-art text embedding models have motivated researchers and practitioners across diverse fields to leverage the power of contextual text embeddings to drive advancements in their respective domains[9–13]. However, the extensive and increasing availability of pre-trained language models has led to the proliferation of potential text embedding models, creating confusion regarding model selection for different use cases[14].

In this work, we present a comprehensive benchmark of pre-trained text embedding models to evaluate their effectiveness in capturing and representing the semantics of textual descriptions related to built assets. Through this evaluation, we aim to identify the strengths and limitations of existing language models in enhancing data alignment practices within the built asset domain. Key contributions include: the public release of six novel datasets derived from two industry-renowned information classification systems (see Methods section), the benchmarking of 24 state-of-the-art models, and the public release of the benchmarking software. The developed datasets, encompassing architectural, structural, mechanical, and electrical subdomains, amount to a total of more than ten thousand data entries across six tasks within clustering, retrieval, and reranking categories. This diversity of the developed datasets enabled us to perform the most comprehensive evaluation in this specialized field, to date. In addition to public availability, our proposed datasets and software resources adhere to the evaluation protocols established by the MTEB[14] framework. This alignment (see Benchmark section) is meant to facilitate future research endeavors and continuous improvements, given MTEB's recognized robustness and utility in both academic and practical applications.

Related work

Earlier text embedding methods such as word2vec[15] and GloVe[6] relied on static word embeddings, i.e., tokenizing text input at word level and assigning each word a single, context-independent numerical representation derived from word co-occurrence statistics in large corpora. Subsequent breakthroughs in deep learning and natural language processing research, particularly with the advent of the transformer architecture[16], significantly expanded text embedding capabilities. Modern language models, pre-trained on variants of the transformer architecture, such as BERT (Bidirectional Encoder Representations from Transformers)[17] and GPT (Generative Pre-trained Transformer)[18], generate embeddings that adapt to surrounding text, providing context-aware interpretations of language[7]. Leveraging pre-trained transformer-based language models, researchers have investigated diverse techniques to improve text representation. From earlier contributions on adapting encoder-based architectures (e.g., BERT) to generate embeddings at sentence[19,20] or paragraph[21] level, to later works on model fine-tuning with novel objectives (e.g., contrastive learning[22]), adapting decoder-based architectures for text embedding[8], or instruction-based fine-tuning[23,24], researchers have been investigating novel ways to refine how linguistic context and domain-specific nuances are incorporated into text representations.

Recently, an increasing volume of research has been conducted to study text embeddings for diverse applications such as database integration[10], biomedicine information management[9], public figure perceptions in social science studies[12], and many others in various specialized domains[11,13,25,26]. Similarly, given that built asset data predominantly exists in the textual form[27], the built environment literature has witnessed an increasing interest in studying pre-trained language models. In particular, an emerging research direction focuses on utilizing these models for information alignment, addressing a key challenge in advancing workflow automation in the built asset information management[28]. Recent studies in this area include the extraction of built asset entities from unstructured sources[29,30], alignment across construction schedules and entity classification taxonomies[28], automated mapping of building information metamodels[31], and the matching of building material information[4]. However, given the novelty of the topic, two important gaps can be observed in the existing literature. First, many of the related studies have been significantly limited in scope, primarily focusing on ad-hoc downstream tasks with small evaluation datasets, which can result in a potentially skewed perspective on the overall domain-specific text understanding of these models[29]. Second, scarce public access to the datasets used in previous works, which undermines the transparency and reproducibility of the reported results.

In addition to the gaps highlighted above, recent research indicates that general-purpose text embedding models often struggle to maintain consistent performance across diverse tasks and domains[7]. In this light,

researchers have increasingly studied rigorous methodologies for text embedding evaluation. While initial efforts focused on general-purpose assessments which predominantly covered high-resource languages such as English[14,32], there has been a growing emphasis on the creation of evaluations tailored to more diverse and inclusive linguistic contexts[33–36]. Moreover, studies in specialized domains such as biomedicine[9,25] or chemistry[36] have highlighted the need for domain-specific text embedding evaluations. This trend, alongside the established robustness of existing evaluation frameworks such as Massive Text Embedding Benchmark (MTEB)[14], motivates us to examine the effectiveness of the state-of-the-art language models in delivering contextually-accurate mappings of domain-specific terminology within the context of built asset information management.

Methods

Data sources

Given the built environment's multidisciplinary nature, the datasets included in the benchmark must encompass an expansive spectrum of sub-domain subjects, including architectural, structural, mechanical, and electrical systems. To ensure a diverse coverage of built products in our benchmark, we carefully examined the selection of data sources used for creating task-specific datasets. A detailed description of the corpus development and data extraction processes is provided below.

The initial step in creating the benchmark's task-specific datasets is the development of a consistent corpus of built products. Based on the requirements of the tasks within our benchmark, the core corpus needed to include the following key information for each product: name or title, description, and corresponding labels (group categories). The two primary sources used to develop the built product corpora are as follows:

Industry Foundation Classes (IFC). Published and maintained by buildingSMART International[37], IFC is an open international data model offering comprehensive digital descriptions of various aspects of building and infrastructure projects. Originally designed to facilitate interoperability and information exchange among different software applications and stakeholders, IFC provides a comprehensive representation of various aspects of built asset entities. We utilize IFC version 4.3.2.0[38], recently approved as an ISO standard (ISO 16739-1:2024).

Uniclass. Developed and maintained by the National Building Specification (NBS)[39], Uniclass is a unified classification system for the built environment. We utilize version 1.33 of the Uniclass Pr Product Table[40]. Uniclass has extensive coverage, encompassing over 8,000 product types, making it one of the most recognized and widely adopted classification systems in the built asset industry.

Data extraction

To create a corpus of products with corresponding names, descriptions, and labels, we undertook the following steps: For Uniclass, we utilize the publicly-available CSV format of the products table[40]. This table originally contained product names and numerical identifiers representing hierarchical product classifications, extending up to three ancestral levels. For example, as depicted in Fig. 1a, the identifier "Pr_20_29_03" (Anchors and components) denotes a fine-grained classification within the "Pr_20_29" (Fastener Products) class, which itself is a subclass of the "Pr_20" (Structure and General Products) primary class. Hence, for each Uniclass record, explicit product categories were required to be inferred from the corresponding numeric identifiers. To automatically extract the explicit textual labels for each product, we developed a script to retrieve and process the table data programmatically and recursively traverse each record and infer the product's membership across the three levels of the parent-child taxonomic hierarchy. Moreover, since the original table does not include product descriptions, we propose a method (detailed in the subsequent subsection) to synthesize a description for each product. We retained only those products that have labels for all three classification levels. After applying this filtering process, the Uniclass corpus comprises 4,234 instances, which remains sufficiently large for our benchmarking purposes.

Regarding the IFC schema, we parse the official schema content by utilizing resources from an open-source Python library[41] that enables programmatic access to IFC entities. Initially, we extracted entities of interest from a JSON-formatted file[42] containing the exhaustive list of IFC entities, their type enumeration, and their definition (derived from IFC's official documentation). After excluding IFC entities with missing descriptions (less than 1% of total "IfcElement" entities), we developed a script to extract each entity's top three parent classes to serve as the product category labels. An analysis of the "IfcProduct" class within the IFC schema indicated that a significant majority of product entities are classified under the "IfcElement" class. Based on this observation, subclasses of "IfcElement" were designated as the highest-level valid label instances. For example, as illustrated in Fig. 1c, enumerations of sensor types (e.g., CO2 sensor) are labeled as "sensor" at the immediate categorical level, with "Distribution Control" and "Distribution" as higher-order product categories. This three-level categorization facilitates a unified structural framework for integrating both IFC and Uniclass records within a single built asset product corpus. In the case of the IFC corpus, in addition to the hierarchical groupings of the entities, we use the domain-specific schemas (e.g., structural, HVAC, building control) derived from IFC's official documentations[38] as an auxiliary source for entity label assignment. These labels were incorporated into the IFC's corpus dataset under the "tag" field. The resulting IFC corpus comprises 977 entities (total of parent entities and type enumerations).

Data augmentation and curation

The process of generating textual descriptions for Uniclass entities is depicted in Fig. 1a. Initial entity descriptions are synthesized by sequentially concatenating the entity's category titles, progressing from the most specific to the most general. An example of the synthesized descriptions is provided in Fig. 1a. These concatenated descriptions are then paraphrased using a generative language model to create more nuanced and natural descriptions, relaxing the text from the rigid template initially employed. We generated paraphrased descriptions

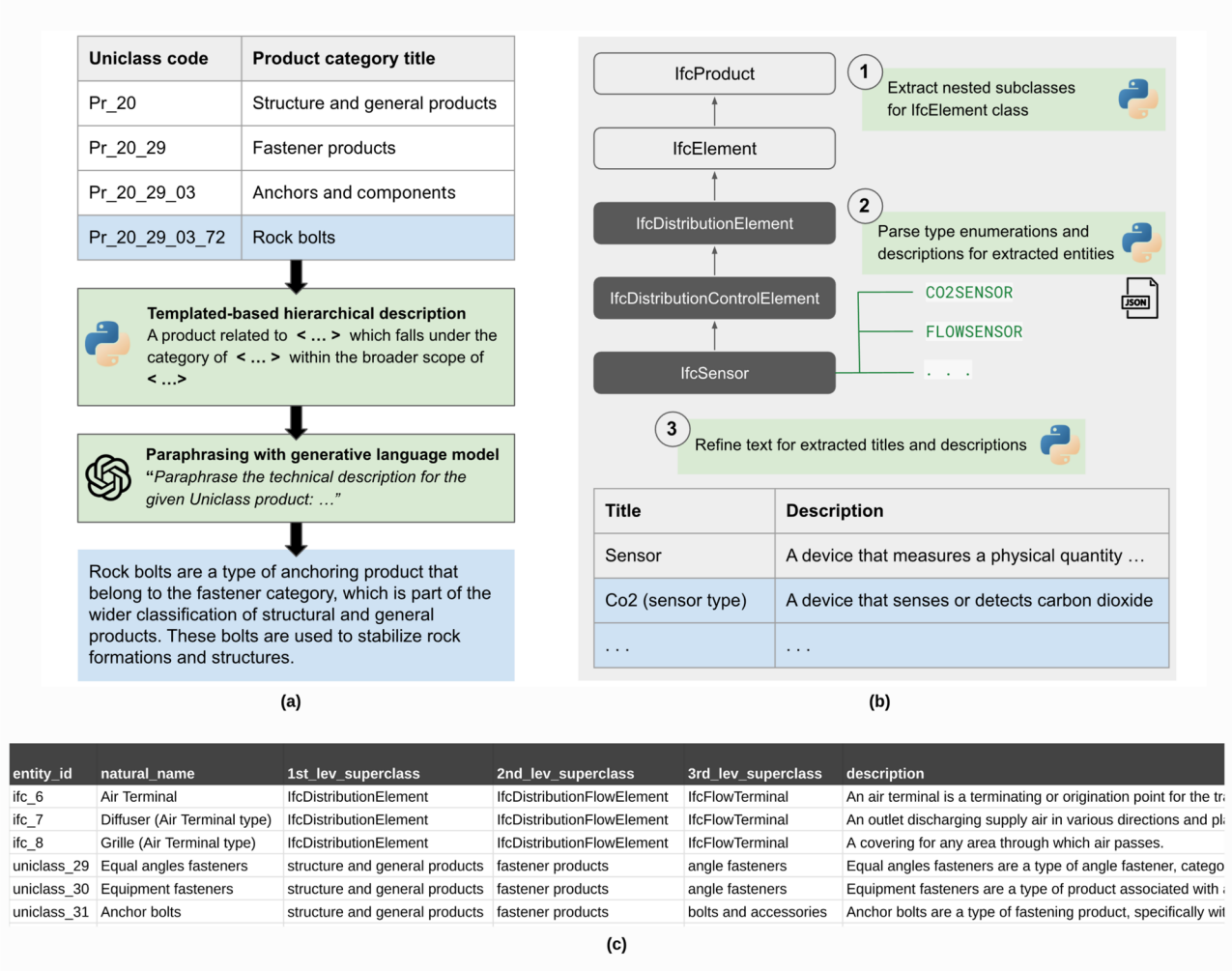

Figure 1. Overview of the main steps in developing the built product corpus: (**a**) Example of extracting categories and synthesizing entity descriptions from raw Uniclass entries; (**b**) Example of hierarchical relation extraction for main entities and their enumerated types from the IFC schema; (**c**) Sample records from the developed corpus, containing product titles, descriptions, and categories with three levels of granularity.

using the most advanced version of the GPT-4 model available at the time of conducting the experiments (gpt-4-turbo-2024-04-09).

Following the official documentation of the GPT-4 model API, two prompt templates were employed. As it can be seen from Fig. 2, the system prompt explains the main task and key instructions to follow during the text generation, and the user prompt template is populated for each record with the product title and initially synthesized description. The system prompt explicitly specifies for the model to act as an expert in Uniclass product classification and emphasizes the preservation of contextual information and technical accuracy during the paraphrasing process.

Although the instructions included within the system prompt were designed to prevent the alteration or addition of facts, it was essential to manually review all generated descriptions due to known potential inaccuracies of generative language models, and the risk of model hallucinations. The review was carried out by two domain experts, each with over ten years of experience in the field. Each expert cross-checked the issues identified by the other, and the final decisions were made based on mutual agreement. Of the total 4,234 paraphrased descriptions, only 16 cases required adjustment by the two reviewers. The low edit rate was expected, given the simplicity of the task (paraphrasing short, well-written English definitions extracted from established industry standards).

To ensure the semantic integrity and syntactic uniformity of the extracted product names and descriptions, a multi-step preprocessing pipeline was implemented. Since the entity names in the IFC schema are originally represented in camel case format, a string normalization procedure was executed to convert them into concrete product names by delimiting constituent words and removing the 'Ifc' prefix (e.g., "IfcHeatExchanger" is converted to "Heat Exchanger"; see examples in Fig. 1b, c). For IFC class enumeration types, where the enumeration name alone might be ambiguous, we append the parent class type in parentheses. For example, the enumeration "WATER", a subclass of "IfcBoilerTypeEnum", is represented as "Water (Boiler Type)" (see examples

System prompt:
You are a helpful assistant who is an expert in "Uniclass" construction products classification. Your task is to paraphrase the technical description for given Uniclass products.
Follow these instructions when paraphrasing from the original description:
- Ensure that all critical information is contextually preserved.
- Ensure the paraphrased description maintains the technical accuracy of the original terms.

User prompt (template):
Write a paraphrased version of the below product description.

Original:
{product_name}: {description}

Paraphrased:

Figure 2. Contents of the system and user prompts employed to paraphrase synthesized Uniclass product descriptions. For each record of the Uniclass corpus dataset, the user prompt template was populated with the record's product title and initial synthesized description.

in Fig. 1c). Following the same logic, we enrich the product descriptions by concatenating the product's name at the beginning of the description for both Unicalss and IFC entities. This step reinforces contextual clarity, as the natural entity names carry significant semantic information. A comprehensive text normalization process was subsequently applied, utilizing Python's regular expression library to identify and rectify syntactic irregularities. This included the detection and removal of special characters (excluding essential punctuation), the normalization of inconsistent whitespace, and the resolution of uppercase words (excluding valid abbreviations). Additionally, a rigorous manual inspection was undertaken. This involved the review and modification of entity descriptions to eliminate inconsistencies, such as notes related to the schema version history or future deprecation notices, thereby ensuring the production of a semantically coherent product corpus.

Sampling

To ensure a robust entity selection when creating task-specific datasets, we employed positive and negative sampling strategies as follows:

Positive sampling. For positive sampling, we adopt a semantic diversity approach with the main objective of selecting samples that belong to the same class (e.g., fastener products) but are semantically diverse. Given a targetted subset of built products, we generate text embeddings for all corresponding text inputs, i.e., product names and descriptions. Embeddings are generated using a state-of-the-art text embedding model ("mxbai-embed-large-v1"[43]). From this set of embeddings, we randomly choose an initial sample as a starting point. Subsequently, we iteratively select additional samples by identifying those that exhibit the *lowest* similarity to the most recently selected sample, as determined by cosine similarity scores, i.e., the cosine of the angle between two embedding vectors. This process repeats until the desired number of samples is achieved. This method ensures that the samples selected for a particular subset (e.g., products of the same category) yield diverse representations within the embedding space by selecting inputs that are semantically dissimilar to the ones already chosen.

Negative sampling. In negative sampling, i.e., selecting samples from different classes (e.g., products not belonging to the fastener category), the objective was to select negative samples that, while belonging to a distinct class (e.g., furnishing product), yielded closer semantic similarity to a given query (a product name or description). We compute the cosine similarities between the query and negative samples using the same embedding model used in the semantic diversity sampling and select samples with the *highest* similarity scores. By selecting more similar candidates as negative samples, the dataset can better benchmark the model's capability to capture the subtle differences between closely related classes. This method, commonly known as hard negative sampling, is particularly effective for evaluations involving fine-grained classifications, such as differentiating between closely related categories in IFC and Uniclass classification hierarchies.

Lastly, irrespective of the sampling method (positive, negative, or plain random selection) being performed, we maximize the coverage of the entire data pool by ensuring that once a sample is selected, it is re-used in another subset only when all samples in the pool have been exhausted. This prevents repeated oversampling of any particular record and helps maintain diversity across the benchmark datasets.

Benchmark

Tasks overview

Evaluating text embeddings across different tasks is crucial for assessing the transferability of their capabilities to various downstream applications. Hence, our proposed benchmark covers three main tasks: clustering, retrieval, and reranking. In addition to domain coverage and cross-task adaptability, evaluating text embedding models requires careful consideration of input text length. To ensure the coverage for varying input lengths, the text entities included in our datasets fall into two categories: (a) sentences, which are derived from product titles/

names, and (b) paragraphs, which are derived from product descriptions/definitions. Accordingly, each task-specific dataset in our benchmark is grouped into one of the following categories:

- *Sentence to Sentence (S2S):* These tasks involve matching or comparing short-form text inputs. The current version of the benchmark uses product titles as sentence input. Examples of real-world scenarios include grouping products based on their titles (clustering-s2s) or retrieving the tiles of similar items based on compact keyword-focused queries (retrieval-s2s).
- *Paragraph to Paragraph (P2P):* These tasks involve matching or comparing long-form text inputs, i.e., product descriptions (which can be concatenated with the product name). P2P evaluations are critical for real-world scenarios that necessitate the processing of longer textual data, which often encompass more complex definitions and detailed specifications, e.g., retrieving similar products from detailed product catalogs.
- *Sentence to Paragraph (S2P):* These tasks involve comparing a short text input (product title) against a longer, more detailed text (product description). S2P tasks examine how well embeddings handle cross-length comparisons, such as ranking the most relevant comprehensive product details based on concise user queries.

Our proposed benchmark follows the MTEB[14] framework for reporting text embedding performance evaluations. In particular, we adhere to MTEB's core rationale and guidelines for defining tasks, preparing datasets for each task, and selecting evaluation metrics. This alignment significantly improves the consistency and comparability of our results with state-of-the-art text embedding evaluation studies across diverse domains. Moreover, the maturity and comprehensiveness of MTEB's software provide flexibility for specialized experimental setups, such as tuning clustering granularity or retrieval depths. The current version of our benchmark covers three types of tasks that are included in MTEB: clustering, retrieval, and reranking. The selection of these tasks is motivated by their correspondence to the practical, multi-stage challenges of data alignment in built asset information management workflows. For example, the clustering task can evaluate an embedding model's ability to perform semantic harmonization. A common related scenario is where practitioners must first group and reconcile inconsistent, informal terminology from various sources before this data can be mapped to a formal classification system (e.g., grouping "Air Handling Unit", "Air Handler", "AHU", and "Rooftop Unit" within subcontractor schedules). Retrieval tasks correspond to common scenarios where asset managers, facility operators, or engineers must reliably find relevant product information or specifications from extensive digital product catalogs. Finally, reranking tasks address the practical need for precision and efficiency. Given a set of potential matches from a retrieval system, a model's ability to prioritize the top most relevant documents is crucial for minimizing the manual verification effort required from domain experts. Each task is described in detail below.

Clustering

Clustering tasks involve grouping similar built products into consistent clusters based on their similarities in textual representation. Our proposed tasks include S2S and P2P categories, where product names and descriptions act as input text for each dataset type, respectively. Each clustering task dataset is comprised of various subsets, covering diverse subdomain subjects and different levels of granularity. For each subset, we use product labels derived from the source classification hierarchies (i.e., IFC and Uniclass) as the ground truth category assignments. To create the subsets within each clustering dataset, we first select a subset of product labels (e.g., equipment, signage, furnishings, fittings) from one of the three levels of product hierarchy, either from one specific corpus or across both IFC and Uniclass corpora. We then apply the previously described diversity-based sampling method to sample product names (for S2S datasets) or descriptions (for P2P datasets) for selected labels.

To ensure the quality of the subsets, we evaluate the baseline scores using two embedding models, one for the upper threshold ("mxbai-embed-large-v1"[43]) and one for the lower threshold ("paraphrase-multilingual-MiniLM-L12-v2"[19]). A subset is included in the dataset only if its score with the upper threshold model is below 0.8 and greater than 1/N with the baseline model, where N is the number of unique labels. The upper and lower thresholds are set to maintain task difficulty and ensure the task performs better than random guessing, respectively. Subsets meeting these criteria are shuffled to eliminate order bias before being added to the dataset.

We compute V-measure scores[44] by training a mini-batch k-means model using vector embeddings, with k set to the number of unique labels in each clustering subset. The V-measure, ranging from 0 to 1 (higher is better), represents the harmonic mean of two distinct metrics: homogeneity and completeness. Here, homogeneity measures the extent to which clusters contain only products from a single category, while completeness indicates how well all products from a given category are grouped into the same cluster. More details regarding the calculation of V-measure can be found in[44].

Retrieval

Retrieval tasks aim to identify relevant documents, i.e., product textual descriptions, in response to a given query. Our proposed retrieval datasets are framed as S2P and P2P tasks, where built asset descriptions serve as the corpus (the documents to be retrieved), and product titles and descriptions act as queries for the S2P and P2P tasks, respectively. In both retrieval and reranking tasks, we use the Uniclass corpus as the searchable dataset, while IFC product titles or descriptions serve as queries. This decision is driven by the observation that the Uniclass corpus contains significantly more records than the IFC one and often includes multiple records mapping to a single IFC entity. Accordingly, the cross-classification mappings between IFC and Uniclass define the ground truth for relevancy. In particular, the query-document relevancy is derived from existing mappings that identify the alignment between IFC and Unicalss entities. Specifically, each Uniclass product record

originally includes a field referencing its equivalent IFC entity. These mappings are validated and published by NBS[39] and can be found in the official Unicalss table release[40].

First, we encode all queries and product descriptions into corresponding embedding vectors. These embeddings are then used to calculate the pairwise similarity between a given query and all product descriptions using cosine similarity. Subsequently, product descriptions included in each retrieval dataset are ranked according to descending cosine similarity scores. Finally, we compute nDCG@10 (Normalized Discounted Cumulative Gain[45] at rank 10) as the primary metric. This score, which can range between 0 and 1 (higher is better), reflects the relevancy of the ranked products based on their positions within the top 10 ranks by applying a logarithmic discount factor to penalize results that appear lower.

Reranking

In our reranking tasks, the aim is to rank a set of product descriptions with reference to their relevance to a product query. Similar to retrieval tasks, reranking tasks are framed as S2P and P2P types, and pairwise similarity between query and product description embeddings is computed based on cosine similarity. Given the structural similarity between reranking and retrieval tasks, the ground truth rationale follows the same approach, i.e., using cross-classification mappings to establish which Uniclass records are relevant to a given IFC-originated query.

The primary distinction between retrieval and reranking tasks lies in their scope and focus. While our retrieval tasks involve ranking the entire product corpus, reranking narrows the focus to a smaller set of positive and negative subsets, which are selected using the methods outlined in the previous section to ensure diversity and difficulty (avoiding very high scores from overfitting) within the dataset. Positive and negative samples are selected using the sampling methods described in the previous section, thereby preserving both dataset diversity and the inherent difficulty of the evaluation task. By concentrating on a smaller and more challenging group of product descriptions, our reranking tasks aim to provide a more fine-grained evaluation of the model's ability to rank relevant items accurately.

Similar to retrieval tasks, we use cosine similarity to compute pairwise similarity between a given query and product descriptions included in corresponding positive and negative sets. Subsequent to ranking the descriptions based on the cosine similarity scores, we compute MAP (Mean Average Precision) as our primary metric. MAP provides an averaged measure of precision across all relevant products, ranging between 0 and 1, with higher values indicating better performance. It is worth noting that retrieval metrics reflect overall ranking quality while reranking metrics focus on how early relevant products appear in the list.

Results

Table 1 provides a comprehensive summary of the dataset statistics across the three main tasks in our benchmark. The unique number of sample entries in our clustering datasets shows that more than half of the samples available from the combined product corpora could pass the quality thresholds explained in the methods section. In the retrieval and reranking task, the same retrieval and reranking document corpus is shared between the subtasks of each task category. This design enables a comparative analysis of model performance on different query types, with S2P focusing on shorter product names and P2P targeting longer product descriptions. We applied a 1:3 positive-to-negative sampling ratio to create a balanced yet challenging evaluation set, ensuring that models must distinguish effectively between relevant and irrelevant documents.

To outline the distinctions between our newly constructed datasets and existing ones, we conducted a thematic semantic similarity comparison between our clustering datasets and those from MTEB benchmark. Using the "stella-en-400M-v5" model, which is the most performant small-sized model in our evaluations (see Table 2), we generated embeddings for 200 randomly selected samples and averaged them within each dataset. Figure 3 depicts the cosine similarity matrix as a heatmap, where darker shades indicate higher content similarity. The high similarity scores between our proposed subtasks confirm strong internal consistency within our benchmark. Moreover, moderate to high similarities with StackExchange, Reddit, and Arxiv datasets reflect thematic overlaps with broader domain content. A discussion of the observed similarities is provided in the next section.

The software used to conduct our benchmarking experiments relied on MTEB[46] (version 1.14.5) and sentence-transformers[47] (version 3.0.1). In our experiments, we evaluated models across a broad range of sizes,

Clustering tasks	No. of subsets	Unique/total samples	Avg. sample length	Total no. of unique labels	Avg. unique label per subset
Clustering-s2s	18	2545/3815	28.04	31	5
Clustering-p2p	20	3067/4577	207.91	35	5
Retrieval tasks	**No. of queries**	**Avg. query length**	**No. of documents**	**Avg. document length**	**No. of document per query (Avg.)**
Retrieval-s2p	977	30.35	2761	312.75	8
Retrieval-p2p	977	128.5	2761	312.75	8
Reranking tasks	**No. of queries**	**Avg. query length**	**No. of positives (unique/total)**	**No. of negatives (unique/total)**	**Avg. samples length**
Reranking-s2p	179	27.89	1253/1253	2281/3759	310.15
Reranking-p2p	179	140.44	1253/1253	2241/3759	309.66

Table 1. Summary of dataset statistics per benchmark task.

Tasks (→)	Clustering		Retrieval		Reranking		Avg.	Param.	MTEB
Models (↓)	s2s	p2p	s2p	p2p	s2p	p2p	-	(mil)	Rank
Pre-trained without task instructions									
gte-base-en-v1.5	48.38	51.83	79.98	59.42	66.54	66.73	62.15	137	39
gte-large-en-v1.5	43.42	51.05	83.32	63.27	**72.76**	70.15	64.00	434	24
bge-base-en-v1.5	43.00	51.78	82.56	61.65	67.01	63.38	61.56	109	43
bge-large-en-v1.5	46.69	52.41	82.60	64.86	68.44	65.47	63.41	335	35
UAE-Large-V1	45.45	49.53	83.32	66.42	70.04	68.53	63.88	335	29
GIST-Embedding-v0	46.43	49.96	82.82	62.78	68.81	65.75	62.76	109	41
GIST-large-Embedding-v0	47.97	47.91	84.01	67.06	69.53	68.03	64.08	335	34
e5-base-v2	42.59	50.24	80.83	61.46	69.11	62.91	61.19	109	64
e5-large-v2	42.11	49.45	81.95	64.63	68.61	64.58	61.89	335	55
multilingual-e5-large-instruct	48.01	52.82	80.35	64.55	67.85	65.90	63.25	560	42
multilingual-e5-small	42.98	48.16	76.38	55.03	64.78	62.34	58.28	118	112
all-MiniLM-L12-v2	42.00	46.52	79.97	58.81	66.20	63.97	59.58	33	117
paraphrase-multilingual-MiniLM-L12-v2	37.60	45.70	69.01	49.90	61.23	59.15	53.77	118	136
gte-base	45.96	51.55	82.91	62.95	68.97	66.26	63.10	109	51
gte-large	48.54	55.24	84.32	66.08	70.94	69.25	65.73	335	47
gte-small	44.31	55.55	82.37	60.55	68.82	65.23	62.80	33	70
Pre-trained with task instructions									
gte-Qwen2-7B-instruct	50.19	62.39	**86.28**	73.20	69.47	67.51	68.17	7069	6
mxbai-embed-large-v1	47.49	52.45	83.51	66.60	70.10	69.66	64.97	335	28
multilingual-e5-large-instruct	48.10	59.43	82.91	64.42	70.53	69.23	65.77	560	42
NV-Embed-v2	**58.61**	**67.34**	85.23	**77.02**	66.67	70.34	**70.87**	7851	1
stella-en-1.5B-v5	53.60	54.57	84.18	71.21	71.57	71.77	67.82	1545	3
stella-en-400M-v5	53.39	55.78	84.60	70.00	69.58	69.36	67.12	435	7
Proprietary embedding APIs									
text-embedding-3-small	49.72	49.72	79.97	65.68	65.33	66.99	62.90	-	58
text-embedding-3-large	49.75	55.48	84.99	75.38	71.93	**72.46**	68.33	-	30

Table 2. Average scores of benchmarked models per task, based on the task-specific metrics mentioned in the task descriptions. The first and second highest scores for each task are highlighted in bold and underlined, respectively. MTEB ranks are sourced from records as of September 21, 2024.

from relatively small models with 33 million parameters to significantly larger models exceeding seven billion parameters. However, due to computational constraints, the majority of models tested have less than one billion parameters. The selected models span various positions on the most recent record of MTEB leaderboard (as of September 21, 2024), ranging from first place (i.e., "NV-Embed-v2"[24]) to 136th place (i.e., "paraphrase-multilingual-MiniLM-L12-v2"). For models that are pre-trained with instruction-based data, we used built-in or recommended prompts as provided in the model card's official web page or associated research papers, when available. For example, "mxbai-embed-large-v1" requires custom prompts only for retrieval and reranking tasks, while "NV-Embed-v2" needs specific task-based prompts for clustering tasks as well. For models without built-in task instructions, we applied a general set of prompts to ensure consistency across tasks (prompts are available at the project's public GitHub repository[48]).

The top-ranked model in our benchmark, "NV-Embed-v2", also holds first place on the latest MTEB leaderboard. However, it does not consistently outperform all other models across all tasks. In fact, a closer examination reveals variability in model size and performance relationship. For example, "gte-small", the smallest model in our evaluation with 33 million parameters, delivers competitive scores, nearly matching the average scores of models ten times its size and even outperforming larger models in specific tasks. Despite the previously reported strong correlation between model size and performance[14], our experiments show that superior performance associated with larger models is only evident at the extreme upper end of the parameter scale. This observation supports the growing emphasis on developing and deploying smaller, more efficient models for both research and real-world applications in this specialized field, particularly for low-latency or edge deployments, and importantly for reducing environmental impacts due to less energy consumption.

Motivated by the hypothesis that existing datasets with similar thematic content would yield comparable performance evaluations, we examined the consistency of relative model performances as follows: Given the observed thematic similarity between our clustering datasets and specific MTEB datasets, particularly "StackExchange" and "Reddit" (see Fig. 3), we compared the rankings of model performance across both our datasets and the selected MTEB datasets. As it can be seen from Table 3, the comparative evaluation of the relative rankings indicates a notable variation in model performances, notably in the case of "multilingual-e5-large-instruct", "gte-small", "stella_en_1.5B_v5", and "text-embedding-3-small". These observed variabilities

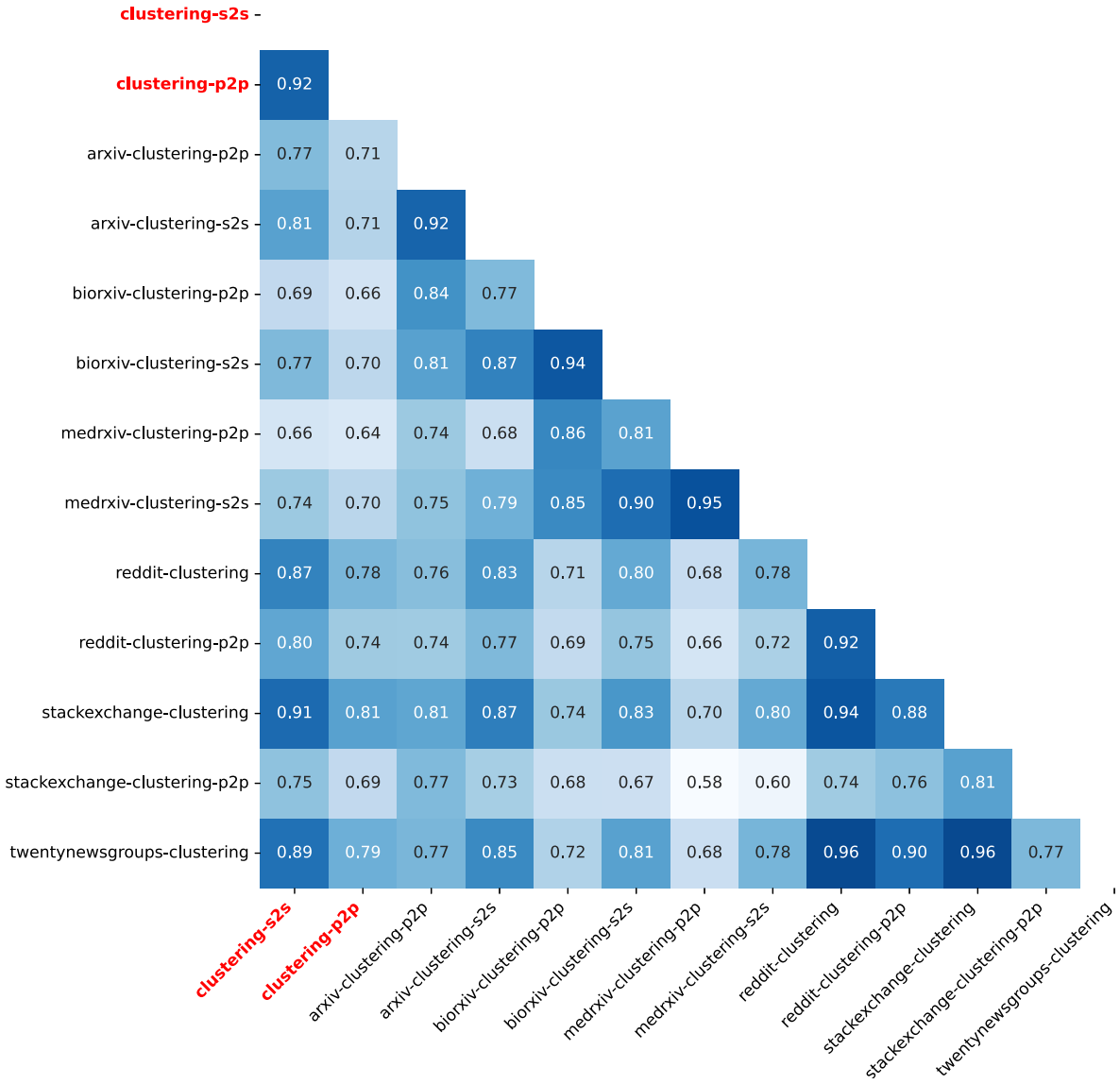

Figure 3. Thematic similarity heatmap between our proposed clustering tasks and those from MTEB. Average embeddings are derived from 200 random samples per dataset, encoded using the "mxbai-embed-large-v1" model[43]. Datasets from our proposed benchmark are highlighted in red.

further highlight the limitations of relying on general-purpose benchmark datasets, even when relatively high thematic similarities are present, underscoring the importance of domain-specific evaluations.

While our benchmarking experiments primarily focused on open-source models, we also included the proprietary text embedding models from OpenAI, both the small and large versions. The inclusion of the proprietary models is motivated by a recent study where closed-source models tend to achieve relatively higher performance when embedding text in underrepresented languages[34]. We hypothesize that built asset text, as an underexplored domain, might be similarly better represented by proprietary models. Notably, text-embedding-3-large ranks second in our benchmark, performing nearly on par with the top-ranked model. In contrast, the smaller model performed more moderately, ranking in the middle of our benchmark. While the former observation aligns with the findings of[34], the latter is in line with the latest MTEB leaderboard results where closed-source commercial embedding APIs generally underperform compared to their open-source counterparts. These observations raise questions about the underlying factors. However, the lack of knowledge about the key characteristics of proprietary models, such as their size and diversity in training data, prevents us from offering a detailed, conclusive account of their relative performance.

Our benchmarking results reveal a notable difference in performance between shorter and longer text inputs in different tasks. In particular, across the board, models consistently show lower performance in the S2S clustering task compared to the P2P one. This observation can be attributed to the limited presence of contextual clues given the significantly short length of the input text in the S2S clustering task (see Table 1). On the other hand, in reranking and retrieval tasks, the majority of the models yield moderately higher scores in S2P tasks.

	NV-Embed-v2	gte-Qwen2-7B-instruct	multilingual-e5-large-instruct	stella_en_400M_v5	gte-small	text-embedding-3-large	gte-large	stella_en_1.5B_v5	mxbai-embed-large-v1	bge-large-en-v1.5	gte-base-en-v1.5	bge-base-en-v1.5	gte-base	gte-large-en-v1.5	e5-base-v2	GIST-Embedding-v0	text-embedding-3-small	UAE-Large-V1	e5-large-v2	multilingual-e5-small	GIST-large-Embedding-v0	all-MiniLM-L12-v2	paraphrase-multilingual-MiniLM-L12-v2
clustering-p2p (ours)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
stackexchange-clustering	1	3	10	4	19	7	9	2	16	14	8	20	11	6	18	15	5	12	17	21	13	22	23
reddit-clustering	4	1	16	3	18	10	6	2	9	15	14	21	12	11	17	13	5	8	19	23	7	20	22

Table 3. Comparison of model rankings across datasets with high thematic similarity (see Fig. 3).

The likely explanation for the latter observation is that the shorter length of the sentences (product names) in S2P tasks can lead to a lower amount of irrelevant information (noise) in the input query. Since product names tend only to encapsulate the critical information about the target product, they can yield more precise and discriminative text (query) representations for similarity matching.

Discussion

Our benchmarking results offer critical insights into the effectiveness of state-of-the-art pre-trained text embedding models in aligning built asset information. One of the key findings of our study is the variability in performance across tasks, even among top-performing models. Our results suggest that model effectiveness is not strongly correlated across model sizes, emphasizing that size alone is not a reliable predictor of model performance in the specialized domain of built asset information management. The interpretation of the relationship between model size and embedding effectiveness is further complicated by the performance gap observed when comparing models pre-trained with and without instruction tuning. Instruction-tuned models showed higher performance in the majority of our benchmark tasks. Considering the larger size of the instruction-tuned models included in our experiments, the latter observation raises an important question for future research: To what extent can instruction-tuning help smaller models adapt to the specialized domain of the built environment? This opens a promising line of investigation into how task-specific training with instruction-based data can better align a model's understanding with the intricate semantics of built asset data, particularly for models with smaller sizes. Finally, in addition to the variability in model performance across different tasks and text input lengths, the results of our comparative examinations highlight the limited transferability of evaluations based on general benchmarks. Our experiments indicate that, even with relatively high thematic similarity, general-purpose benchmarks remain inadequate in capturing the unique semantic complexity and contextual dependencies present in the textual descriptions of the built asset.

The above-mentioned points highlight the critical need for tailored benchmarking datasets to examine the effectiveness of various domain adaptation strategies in this field of research. Our work contributes to the body of research by laying a robust foundation for future evaluations and providing a benchmark that is carefully constructed to reflect the complexities of built asset data. Our proposed datasets cover diverse subdomains and exhibit varying levels of granularity, mirroring real-world scenarios where built products are required to be mapped across various data dictionaries. The datasets can be used not only for evaluating new or fine-tuned text embedding models for cross-mapping built asset data but also as a contextually rich text corpus to support the training of task-specific language models for other downstream tasks, such as information extraction. Finally, this work contributes to the broader discourse on the transferability of the general-purpose language models' capabilities by focusing on built asset data as a representative example of niche and underexplored domains.

Given the following limitations, our findings should be interpreted with caution. One key limitation of our study can be attributed to the data sources that were utilized for corpus development. In particular, identifying sources that were both of high quality and could be redistributed as public datasets was significantly challenging. Although the two sources used in this study, i.e., IFC and Uniclass, are authoritative and collectively cover a broad range of built asset products, they only represent a subset of the extensive classification systems used across the industry, which can potentially restrict the breadth and complexity of the domain captured in our datasets. Moreover, the textual elements extracted from these sources are notably compact and dense with technical descriptions, which differs from the more varied, lengthier, and often noisy nature of real-world data. Notably, the average character lengths of our datasets (see Table 1) are comparatively shorter than those found in large-scale benchmarks (see MTEB[14] and MMTEB[33]), which may limit the interpretability of model performance for longer text inputs. Additionally, despite the manual review of the synthesized descriptions, the developed Uniclass corpus may contain subtle semantic shifts due to paraphrasing. Finally, the text sources used in our work are exclusively in English, limiting the generalizability of our findings to other languages. In this light, the proposed benchmark can benefit from future works attempted at adding large-scale datasets that prioritize diverse text sources across sub-domains, languages, and lengths. Expanding the dataset size would facilitate the creation of training and validation splits, thereby enabling the inclusion of additional benchmark tasks that rely on supervised learning methods.

Conclusion

This study presents a comprehensive evaluation of pre-trained text embedding models for built asset information alignment, proposing six tailored datasets across clustering, retrieval, and reranking tasks. The benchmarking of state-of-the-art models revealed performance variability, underscoring the complexity of domain-specific evaluations and the limitations of relying solely on general benchmarks. Notably, the reported observations showed that smaller models occasionally matched or outperformed larger counterparts, highlighting the importance of training strategy and data quality over model size. A key finding of the study is the limited transferability of general-purpose benchmark evaluations to the built asset domain, even when high thematic similarities exist with texts from other domains. Additionally, the results highlight the promising potential of instruction-tuned models in representing domain-specific terminology. Future research should extend these findings by incorporating larger and more diverse multilingual datasets, study the underlying causes of variations observed against general benchmarks, and further investigate the comparative effectiveness of the state-of-the-art domain adaptation strategies. Moreover, evaluating embeddings alongside non-text modalities that are ubiquitous in built asset documentation (e.g., 2D drawings, images, 3D models) remains a promising area for future studies. Our published datasets and software aim to facilitate future research, supporting ongoing advancements in the automated alignment of built asset information.

Data availability

The developed datasets and codes are openly accessible at the following GitHub repository: https://github.com/mehrzadshm/built-bench-paper and Hugging Face page: https://huggingface.co/mehrzad-shahin. All materials are licensed under the Creative Commons Attribution-NoDerivatives 4.0 International License (CC BY-ND 4.0). Any future updates, including references to additional data and relevant resources, will be incorporated into the designated GitHub repository.

Received: 23 October 2024; Accepted: 25 June 2025
Published online: 04 July 2025

References

1. Love, P. E. & Matthews, J. The 'how' of benefits management for digital technology: From engineering to asset management. *Autom. Constr.* **107**, 102930 (2019).
2. Moretti, N., Xie, X., Merino Garcia, J., Chang, J. & Kumar Parlikad, A. Federated data modeling for built environment digital twins. *J. Comput. Civ. Eng.* **37**, 04023013 (2023).
3. buildingSmart International. buildingsmart data dictionary (bsdd) (accessed 24 June 2024, 2024). https://www.buildingsmart.org/users/services/buildingsmart-data-dictionary/.
4. Forth, K., Berggold, P. & Borrmann, A. Domain-specific fine-tuning of LLM for material matching of bim elements and material passports. In *Proc. of 2024 ASCE International Conference on Computing in Civil Engineering* (2024).
5. Roberts, C. J., Pärn, E. A., Edwards, D. J. & Aigbavboa, C. Digitalising asset management: Concomitant benefits and persistent challenges. *Int. J. Build. Pathol. Adapt.* **36**, 152–173 (2018).
6. Pennington, J., Socher, R. & Manning, C. D. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* 1532–1543 (2014).
7. Lee, J. *et al.* Gecko: Versatile text embeddings distilled from large language models (2024). arXiv:2403.20327.
8. BehnamGhader, P. *et al.* Llm2vec: Large language models are secretly powerful text encoders (2024). arXiv:2404.05961.
9. Zhang, Y., Chen, Q., Yang, Z., Lin, H. & Lu, Z. Biowordvec, improving biomedical word embeddings with subword information and mesh. *Sci. Data* **6**, 52 (2019).
10. Cappuzzo, R., Papotti, P. & Thirumuruganathan, S. Creating embeddings of heterogeneous relational datasets for data integration tasks. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data* 1335–1349 (2020).
11. Wilkho, R. S., Chang, S. & Gharaibeh, N. G. FF-BERT: A BERT-based ensemble for automated classification of web-based text on flash flood events. *Adv. Eng. Inform.* **59**, 102293 (2024).
12. Cao, X. & Kosinski, M. Large language models know how the personality of public figures is perceived by the general public. *Sci. Rep.* **14**, 6735 (2024).
13. Rouhizadeh, H. et al. A dataset for evaluating contextualized representation of biomedical concepts in language models. *Sci. Data* **11**, 455 (2024).
14. Muennighoff, N., Tazi, N., Magne, L. & Reimers, N. Mteb: Massive text embedding benchmark (2022). arXiv:2210.07316.
15. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S. & Dean, J. Distributed representations of words and phrases and their compositionality. *Adv. Neural Inf. Process. Syst.* **26** (2013).
16. Vaswani, A. et al. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **30** (2017).
17. Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K. 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)* 4171–4186 (2019).
18. Brown, T. et al. Language models are few-shot learners. *Adv. Neural. Inf. Process. Syst.* **33**, 1877–1901 (2020).
19. Reimers, N. & Gurevych, I. Sentence-BERT: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing* (Association for Computational Linguistics, 2019).
20. Cer, D. *et al.* Universal sentence encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations* 169–174 (2018).
21. Karpukhin, V. et al. Dense passage retrieval for open-domain question answering. In *EMNLP*, vol. 1, 6769–6781 (2020).
22. Gao, T., Yao, X. & Chen, D. Simcse: Simple contrastive learning of sentence embeddings (2021). arXiv:2104.08821.
23. Su, H. *et al.* One embedder, any task: Instruction-finetuned text embeddings (2022). arXiv:2212.09741.
24. Lee, C. *et al.* Nv-embed: Improved techniques for training LLMS as generalist embedding models (2024). arXiv:2405.17428.
25. Rasmy, L., Xiang, Y., Xie, Z., Tao, C. & Zhi, D. Med-BERT: Pretrained contextualized embeddings on large-scale structured electronic health records for disease prediction. *NPJ Digit. Med.* **4**, 86 (2021).
26. Ostendorff, M. *et al.* Evaluating document representations for content-based legal literature recommendations. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Law*, 109–118 (2021).
27. Wu, C. et al. Natural language processing for smart construction: Current status and future directions. *Autom. Constr.* **134**, 104059 (2022).
28. Jung, Y., Hockenmaier, J. & Golparvar-Fard, M. Transformer language model for mapping construction schedule activities to uniformat categories. *Autom. Constr.* **157**, 105183 (2024).
29. Shahinmoghadam, M., Kahou, S. E. & Motamedi, A. Neural semantic tagging for natural language-based search in building information models: Implications for practice. *Comput. Ind.* **155**, 104063 (2024).
30. Zhou, J. & Ma, Z. Named entity recognition for construction documents based on fine-tuning of large language models with low-quality datasets. *Autom. Constr.* https://doi.org/10.1016/j.autcon.2025.106151 (2025).
31. Wang, Z., Bergés, M. & Akinci, B. Pre-trained language model based method for building information model to building energy model transformation at metamodel level. In *ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction*, vol. 41, 17–25 (IAARC Publications, 2024).
32. Thakur, N., Reimers, N., Rücklé, A., Srivastava, A. & Gurevych, I. Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models (2021). arXiv:2104.08663.
33. Enevoldsen, K. *et al.* Mmteb: Massive multilingual text embedding benchmark (2025). arXiv:2502.13595.
34. Enevoldsen, K., Kardos, M., Muennighoff, N. & Nielbo, K. L. The scandinavian embedding benchmarks: Comprehensive assessment of multilingual and monolingual text embedding (2024). arXiv:2406.02396.
35. Zhu, D. *et al.* Longembed: Extending embedding models for long context retrieval (2024). arXiv:2404.12096.
36. Kasmaee, A. S. *et al.* Chemteb: Chemical text embedding benchmark, an overview of embedding models performance and efficiency on a specific domain (2025). arXiv: 2412.00532.
37. buildingsmart international (accessed 24 June 2024, 2024). https://www.buildingsmart.org/.
38. buildingSmart International. Ifc 4.3 documentation (accessed 24 June 2024, 2024). https://standards.buildingsmart.org/IFC/RELEASE/IFC4_3/.
39. NBS. National building specification (accessed 24 June 2024, 2024). https://www.thenbs.com/.
40. NBS. Uniclass (accessed 24 June 2024, 2024). https://uniclass.thenbs.com/.

41. Ifcopenshell (accessed 24 June 2024, 2024). https://github.com/IfcOpenShell/IfcOpenShell/.
42. Ifcopenshell (accessed 24 June 2024, 2024). https://github.com/IfcOpenShell/IfcOpenShell/tree/v0.8.0/src/ifcopenshell-python/ifcopenshell/util/schema.
43. Li, X. & Li, J. Angle-optimized text embeddings (2023). arXiv:2309.12871.
44. Rosenberg, A. & Hirschberg, J. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 410–420 (2007).
45. Järvelin, K. & Kekäläinen, J. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.ms (TOIS)* **20**, 422–446 (2002).
46. MTEB (GitHub repository) (2024). https://github.com/embeddings-benchmark/mteb.
47. sentence-transformers (GitHub repository) (2024). https://github.com/UKPLab/sentence-transformers.
48. Shahinmoghadam, M. built-bench-paper (GitHub repository) (accessed 20 October 2024, 2024). https://github.com/mehrzadshm/built-bench-paper.

Author contributions

M.S.: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data Curation, Writing - Original Draft, Writing - Review & Editing, Visualization; A.M.: Conceptualization, Resources, Writing - Review & Editing, Supervision, Project administration, Funding acquisition.

Declarations

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to M.S. or A.M.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.