applied sciences

MDPI

Article

Towards a Method to Enable the Selection of Physical Models within the Systems Engineering Process: A Case Study with Simulink Models

Eduardo Cibrián [1,*], Jose María Álvarez-Rodríguez [1], Roy Mendieta [2] and Juan Llorens [1]

[1] Computer Science and Engineering Department, Carlos III University of Madrid, 28911 Leganés, Spain; joalvare@inf.uc3m.es (J.M.Á.-R.); llorens@inf.uc3m.es (J.L.)
[2] The REUSE Company, 28919 Leganés, Spain; roy.mendieta@reusecompany.com
* Correspondence: ecibrian@inf.uc3m.es

Abstract: The use of different techniques and tools is a common practice to cover all stages in the development life-cycle of systems generating a significant number of work products. These artefacts are frequently encoded using diverse formats, and often require access through non-standard protocols and formats. In this context, Model-Based Systems Engineering (MBSE) emerges as a methodology to shift the paradigm of Systems Engineering practice from a document-oriented environment to a model-intensive environment. To achieve this major goal, a formalised application of modelling is employed throughout the life-cycle of systems to generate various system artefacts represented as models, such as requirements, logical models, and multi-physics models. However, the mere use of models does not guarantee one of the main challenges in the Systems Engineering discipline, namely, the reuse of system artefacts. Considering the fact that models are becoming the main type of system artefact, it is necessary to provide the capability to properly and efficiently represent and retrieve the generated models. In light of this, traditional information retrieval techniques have been widely studied to match existing software assets according to a set of capabilities or restrictions. However, there is much more at stake than the simple retrieval of models or even any piece of knowledge. An environment for model reuse must provide the proper mechanisms to (1) represent any piece of data, information, or knowledge under a common and shared data model, and (2) provide advanced retrieval mechanisms to elevate the meaning of information resources from text-based descriptions to concept-based ones. This need has led to novel methods using word embeddings and vector-based representations to semantically encode information. Such methods are applied to encode the information of physical models while preserving their underlying semantics. In this study, a text corpus from MATLAB Simulink models was preprocessed using Natural Language Processing (NLP) techniques and trained to generate word vector representations. Then, the presented method was validated using a testbed of MATLAB Simulink physical models in which verbalisations of models are transformed into vectors. The effectiveness of the proposed solution was assessed through a use case study. Evaluation of the results demonstrates a precision value of 0.925, a recall value of 0.865, and an F1 score of 0.884.

Keywords: reuse models; physical models; word embedding; semantic representation; similarity algorithm

check for updates

Citation: Cibrián, E.; Álvarez-Rodríguez, J.M.; Mendieta, R.; Llorens, J. Towards a Method to Enable the Selection of Physical Models within the Systems Engineering Process: A Case Study with Simulink Models. *Appl. Sci.* **2023**, *13*, 11999. https://doi.org/10.3390/app132111999

Academic Editor: Jose Machado

Received: 9 October 2023
Revised: 25 October 2023
Accepted: 30 October 2023
Published: 3 November 2023

1. Introduction

Model-based Systems Engineering (MBSE [1]) has gained traction in the Systems Engineering discipline as a comprehensive methodology to unify techniques, methods, and tools for the design and implementation of cyber–physical systems [2,3] through different types of models [4]. Different types of system artefacts are produced as a kind of model through an engineering process, method, task, or activity. Considering that a model is a type of software artefact, model reuse can be defined as a process that systematically specifies, produces, classifies, retrieves, and adapts models for use in future engineering designs. This simple and powerful definition was introduced in the software reuse discipline [5] to overcome the problem of software failures, increase the productivity of engineers, improve the quality of engineering designs, and create a cost-efficient engineering environment. Nevertheless, challenges related to limited model reuse encompass both technical and non-technical aspects, as indicated in [6]: (1) economic, organisational, educational, and psychological factors and (2) a lack of modelling standards, reusable component libraries, and suitable tools that can enhance reuse and promote interoperability.

In the context of technical challenges, software engineering methodologies have undergone extensive examination, as exemplified in [7]. This examination aims to reinforce the enduring principles of software reuse, as outlined in [8], encompassing abstraction, selection, specialisation, and integration. Specifically, the essential feature for any reuse technique can be considered as an abstraction that specifies when an artefact can be reused and how it should be reused, effectively managing the intellectual complexity of a software artefact. The discovery of software artefacts, including their representation, storage, classification, location, and comparison, is referred to as selection. The set of parameters and transformations necessary for reusing a software artefact is encompassed by specialisation, while the capability of software systems to communicate, collaborate, and exchange data to create a combined software system is denoted as integration.

However, the selection of an adequate knowledge representation paradigm for reusing any piece of information persists, as a suitable representation format (and syntax) can be reached in several ways (e.g., ontologies, rules, and knowledge graphs [9]), and there are several standardised languages to choose from, such as the Systems Modelling Language (SysML), Unified Modelling Language (UML), or Modelica. Obviously, different types of knowledge, including inference mechanisms, require different types of representation. Any piece of information must be structured and stored in such a way as to support other application services; moreover, in order to foster a collaborative Model-Based Systems Engineering (MBSE) environment [10], it is necessary to (1) establish a seamless communication system among the various tools used in the development lifecycle, thereby easing access to data and information; (2) provide a method for accurately representing system artefacts and their underlying semantics, e.g., logical/descriptive vs. physical/analytical models; and (3) empower engineering teams to efficiently search for and customise existing engineering designs.

In this work, we introduce a method to support the process of reusing the knowledge embedded within MBSE environments, with a special focus on physical models such as Simulink models. This solution allows users to represent and retrieve information regarding specific attributes within a given set of models, allowing them to perform queries (for instance, "Which electrical models contain an AC voltage source rated at 100 V?"). To achieve this, we have designed and implemented a method that harnesses word embeddings for the representation, indexing, and retrieval of information. The initial dataset was created from Simulink documentation and samples, then further enriched using Wikidata [11] as an external dataset. Natural Language Processing techniques are applied to process the output text and build a word vector representation. To validate the presented approach, a case study with Simulink models was conducted to show the ability of the system to (1) represent physical models as vectors using the trained model and (2) perform a set of queries against this dataset of vectors. In our evaluation, we used the precision, recall, and F1 values as performance metrics to assess the retrieval system.

The rest of this paper is organised as follows. Related works are presented in Section 2, where we discuss previous works in the field of reuse models that form the basis of the present research. Section 3 presents the implemented solution along with a description of the similarity algorithm. Section 4 describes the validation process, including the methodology, results, and analysis of the results. Finally, Section 5 summarises the main conclusions and outlines future research directions.

2. Related Work

The semantic retrieval of physical models is the main objective of this work. Although many studies have developed reuse techniques focusing on system artefact representation, integrating more advanced methods based on artificial intelligence techniques remains a challenge [12].

The continuous evolution of information representation techniques for searching aims to enhance the accuracy and relevance of retrieval systems in order to provide users with more effective experiences. Notable approaches include deep learning models for text representations, which are capable of capturing high-level information; however, these demand extensive labelled data [13]. Additionally, the use of graphs and structured knowledge allows semantic relationships to be captured [14], although this usually requires external data sources [15].

In the context of text representation, recent studies have seen the use of word embeddings as a technique to encode texts as dense low-dimensional vectors [16]. These vectors represent the semantic and syntactic relationships between words, and are learned through unsupervised training on a text corpus such has Word2Vec [17]. Word embeddings can be used to improve the accuracy of tasks such as text classification [18], sentiment analysis [19], and machine translation [20]. In order to retrieve information, this approach has been presented in [21] to calculate the similarity between a query table and a set of tabular datasets, although the implemented algorithm does not consider the similarity with the input text pattern and does not apply preprocessing to the input text, as there is only one term in each cell of the table.

The application of Large Language Models (LLMs) such as OpenAI GPT-x, Google Bard, and Meta Llama to natural language processing tasks is a complete game-changer in the text processing arena; these models use transformers [22] as an underlying architecture to build relationships [23] between tokens derived from raw texts. Although these models can probabilistically understand and produce text, and have been used to perform tasks such as classification, content summarisation, entity recognition, question answering, translation, etc., they lack the ability to perform reasoning without the support of external tools; even more importantly for a retrieval system, they cannot trace the output to an input.

In the context of model reuse, ref [24] introduced an approach for representing and retrieving Computer-Aided Design (CAD) models. This method involves the implementation of a mapping function designed to correlate the features of various CAD models. Additionally, in the domain of model representation, prior work is exemplified in [25,26]. In these studies, the authors proposed a solution for representing components within models using a concept referred to as Physical Model Mappable Element (PMME). A PMME serves as a means to capture information in a manner that transcends specific programming languages, thereby enhancing cross-compatibility and language-independence. In [26], the concept of a metamodel paved the way for demonstrating the feasibility of transforming models to achieve the objective of reuse. The main limitation of these works is that it is necessary to define an ontology that contains the domain terminology and semantics in order to be able to reuse models. This requires significant effort, and there is no guarantee of all the terms that may be used in a query being included in the domain knowledge.

Recently, interoperability-based approaches have been widely studied and developed to define interoperable engineering environments in which data, metadata, and information can be accessed through a standard service interface. Initiatives such as the family of ISO 10303-STEP (Standard for the Exchange of Product Model Data) standards, e.g., ISO 10303-243:2021 [27] "Industrial automation systems and integration—Product data representation and exchange", or the Open Services for Life-cycle Collaboration (OASIS OSLC) specifications, are focused on promoting the implementation of technical engineering processes through an interoperable framework. These initiatives promote the creation of federated environments of services (tools) by defining collaborative engineering ecosystems through data shapes or schemes that serve as contractual agreements for accessing information resources. Both the ISO 10303-STEP and OASIS OSLC families utilise the Representational State Transfer (REST) software versionarchitecture style to manage publicly represented information resources, which can be exchanged in formats such as JSON, XML, or even the Resource Description Framework (RDF). For instance, the specification of the OSLC Knowledge Management (KM) domain represents both the data and metadata of a system artefact [28]. Similarly, the latest version of the SysMLV2 API (Systems Modelling Application Programming Interface and Services specifications) follows this approach by defining a REST API for consuming SysML models as key-value documents. Although these approaches have made great efforts to define an engineering environment as a set of standardised federated set of services, they are mainly focused on metadata exchange, and have only partially addressed the challenges around enabling reuse of system artefacts. In the case of OSLC, the implementations may offer a kind of interface to perform SPARQL-like queries over the indexed metadata without the possibility of properly processing the internal entities and relationships. Other efforts, such as the Open Model-Based Engineering Environment (OpenMBEE) and the Model Management System (MMS) tool, do offer the possibility of accessing specific elements. However, designing the required SPARQL-like queries to gather the proper information resources and deciphering their internal representation are challenging tasks.

In summary, the state of the art highlights the challenges and advancements in the field of semantic retrieval of physical models and information representation techniques for building retrieval systems; while various works have focused on system artefact representation, the integration of more advanced artificial intelligence techniques, such as deep learning models for text representations and word embeddings, remains a challenge. Word embeddings offer a method of encoding texts as dense vectors to capture the semantic and syntactic relationships between words, and have found applications in text classification, sentiment analysis, and machine translation. However, existing approaches may not comprehensively consider the similarity with input text patterns.

3. Proposed Method for Representing and Searching Physical Models

In the context of the semantic reuse of physical models, we have designed a five-step method was (see Figure 1) to illustrate the relevant aspects involved in moving from a domain-specific representation of a physical model to a word-vector representation used to implement a retrieval system. Our main objective was to establish a structured graphical representation of physical models able to encompass model elements and their relationships within a graph. Subsequently, this graphical representation formed the baseline for the generation of detailed textual descriptions of these models. This approach enables queries that are based on metadata while additionally serving as content, thereby allowing the retrieval of information based on the inherent content and semantic concepts of the model.

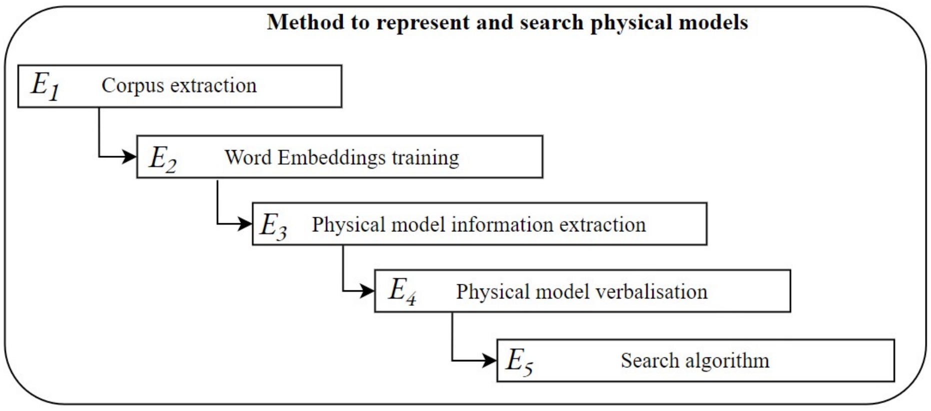
Method to represent and search physical models
E_1 Corpus extraction
E_2 Word Embeddings training
E_3 Physical model information extraction
E_4 Physical model verbalisation
E_5 Search algorithm

Figure 1. Stages of the proposed method for representing and searching physical models.

3.1. Corpus Extraction

The first step involves the creation of an initial corpus of documents, which is a pivotal stage in the development of a word embedding model. The primary goal of this model is to represent the terms associated with physical models within a vector space. To establish this corpus, it is necessary to start with a baseline of terms that can be commonly found in these physical models. This foundational terminology can be sourced from official documentation or manuals that encompass terminology related to physical models. However, in certain cases official documentation or manuals may not offer a sufficiently extensive corpus. Consequently, it becomes essential to augment this foundational information with data from other sources.

In the case of reusing Simulink physical models, the names of the elements of the physical models from Simulink's official documentation were used as input, and similar elements were searched in Wikidata. Specifically, for each term t_i in the input set A (where $t_i \in A$), a new output set of terms B is obtained from the labels, as shown in Figure 2. For each t_i, a document is obtained with the definition extracted from the Encyclopædia Britannica, which becomes part of the training corpus C. In the next iteration, the output set B becomes the input set A. In total, a set of 9620 terms was discovered in this process.

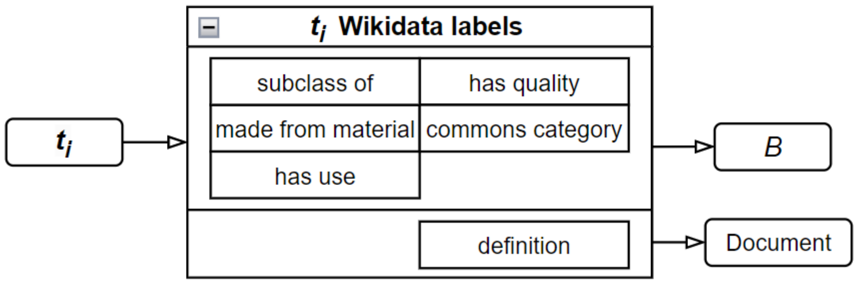

Figure 2. Extraction of the corpus to obtain labels from Wikidata concepts.

3.2. Word Embeddings Training

Word embeddings play a crucial role in modern natural language processing and comprehension by representing and extracting semantic information from textual data. These embeddings serve as the fundamental framework for capturing relationships between words and phrases, thereby enhancing the understanding of machines of human language. In the context of this work, our attention is directed towards the second phase of the methodology, where we harness word embeddings. This step is designed to further enhance the comprehension and representation of terminology and semantics within physical models.

Pretrained models are typically built upon extensive amounts of text data, frequently sourced from references such as Wikipedia. This pre-existing knowledge can substantially enhance the quality and effectiveness of forthcoming tasks. However, when retraining a word embeddings model with a new vocabulary it is important to emphasise that a significant reshaping of word vectors takes place; this adaptation is crucial to ensuring the alignment of word vectors with domain-specific semantics.

1. **Data Collection:** in this initial step, we employ a pretrained model, denoted as M and available from [29]. This model is typically constructed using a substantial corpus, such as the 2021 Wikipedia dump. This corpus forms the foundation for the subsequent stages of the process.
2. **Corpus Preprocessing:** the text corpus C obtained in the prior stage undergoes a series of essential preprocessing steps. These steps are vital for cleaning and preparing the data for further training. Preprocessing can include operations such as tokenisation, which involves breaking text into individual words or tokens; normalisation, which often entails converting text to singular to ensure consistent representation; and the removal of stopwords, which are common words that do not significantly

contribute to the semantic meaning of the text. These preprocessing steps serve to enhance the quality of the input data and ensure the effectiveness of the subsequent training process.

3. **Training with Pretrained Model:** with the preprocessed corpus C at hand, the next phase involves training the word vectors in the pretrained model M. This training process necessitates adjusting the existing word vectors within the pretrained model to better reflect the specific vocabulary and semantics relevant to the given application or domain. The result of this phase is the creation of a new word embeddings model, denoted as $M+$.
4. **Generating New Word Embeddings:** the outcome of the training process in the previous step is a set of word vectors that have been tailored to align with the new terminology and semantics derived from the particular corpus C associated with Simulink models. These adapted word vectors now closely correspond with the domain-specific context, rendering them better suited for tasks such as text comprehension and interpretation related to Simulink models.

3.3. Physical Model Information Extraction

In this step of the methodology, the main objective is access and representation of physical models in the form of a knowledge graph. To do this, the development of a component with three distinct layers is required (see Figure 3): the initial layer, the model layer, assumes the role of extracting all the information embedded within the model; the representation layer creates the graph representation; and the terminology layer is in charge of populating this graph with terms and properties from the models. This structured three-layered approach is essential to obtaining an accurate representation of all elements within the model, as it effectively translates the underlying details of the physical models into a coherent and machine-processable knowledge graph:

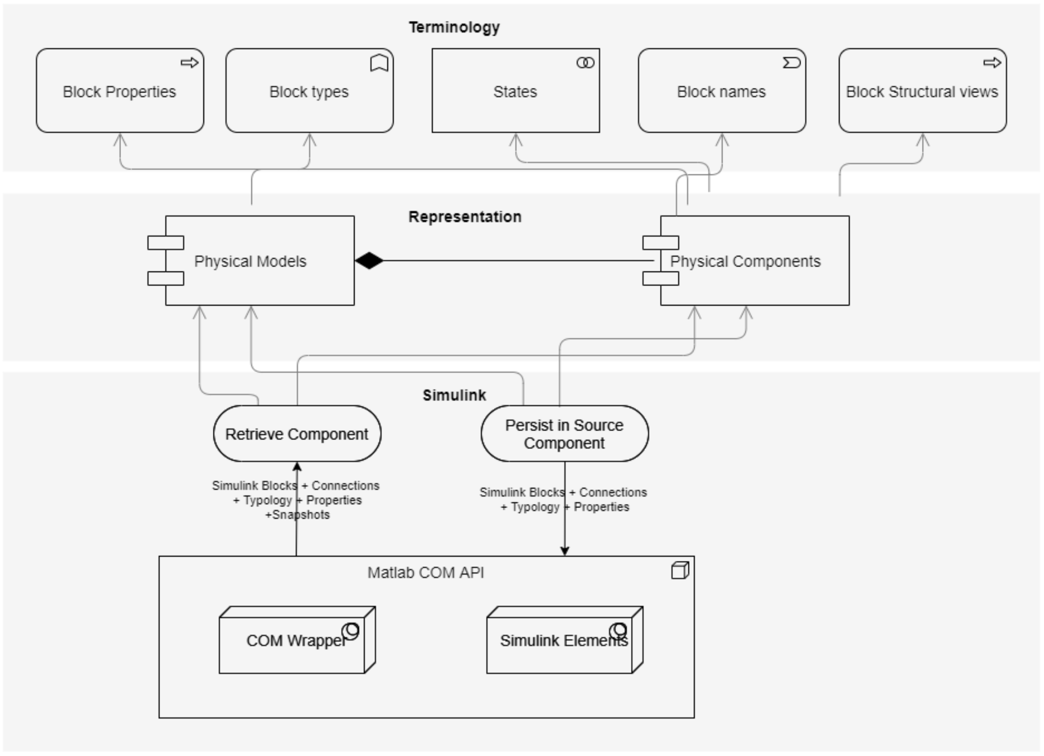

Figure 3. Architecture of semantic interpretation component for Simulink models.

1. **Model Layer:** this layer represents a critical component that enables interaction with the different modelling frameworks and platforms. It facilitates the retrieval of relevant data from physical models, encompassing elements such as blocks, systems, subsystems, properties, and even visual representations. Within this layer, a connector must be customised to harness platform-specific technologies and Application Programming Interfaces (APIs), facilitating the retrieval of data associated with the models. Subsequently, these data are processed and organised in accordance with the data model established within the rendering layer. In the case of Simulink, this layer facilitates all interactions with MATLAB Simulink. It utilises COM technology to retrieve information about the elements that constitute the physical models, including blocks, systems, subsystems, properties, and even image snapshots. Within this layer, two components were developed that leverage the *MATLAB interop* engine [30] to execute commands from the MATLAB console and perform Simulink model component recovery operations. The retrieved information is then processed and represented in the data model defined in the rendering layer.
2. **Representation Layer:** this layer creates a connection between the underlying object model of the source tool and a structured data shape based on a semantic graph known as Mappeable Elements (MEs). MEs offer a standardised representation [28] of the model artefacts, decoupling the representation of entities from the specific object model of the source tool. In the case of Simulink, this layer transforms the Simulink object model into a set of MEs. This representation facilitates the creation of an underlying knowledge graph that can integrate information from different sources.
3. **Terminology Layer:** this layer facilitates the extraction of terminology from various information sources. In the case of Simulink models, it retrieves component names, component types, properties, and even images of the subsystems.

This architecture has been designed and developed in previous works [28] using the Visual Studio .NET 2019 Framework 4.8, and provides a comprehensive framework for the semantic interpretation of Simulink physical models. The model layer ensures smooth communication with MATLAB Simulink, facilitating the extraction of comprehensive information regarding the elements within the physical model. The representation layer plays a critical role in transforming the Simulink object model into a standardised internal data structure using semantic graphs. This transformation ensures a uniform representation across various sources of information, simplifying the integration of knowledge from different tools and frameworks. Finally, the vocabulary layer aids in the extraction and organisation of pertinent terminology from Simulink models, enhancing the overall process of semantic interpretation.

3.4. Physical Model Verbalisation

When the physical model has been represented as a knowledge graph, the next step involves the transformation of this representation into a human-readable text. This textual description encapsulates comprehensive details about the model, its constituent elements, and the relationships among them. This process not only provides a concise summary of the model's contents, it facilitates content-based searches, thereby allowing users to query and retrieve specific information based on the generated descriptions.

Figure 4 illustrates a representation of information from a physical model in a graph. Nodes within the graph contain detailed information and properties of specific components, while the relationships between nodes represent the connections and associations between these components. In physical models, these relationships could include "connected to" or "contains" along with different associated properties. To transform this information represented in the graph into text, the pattern P_1 is used, as shown in Figure 4.

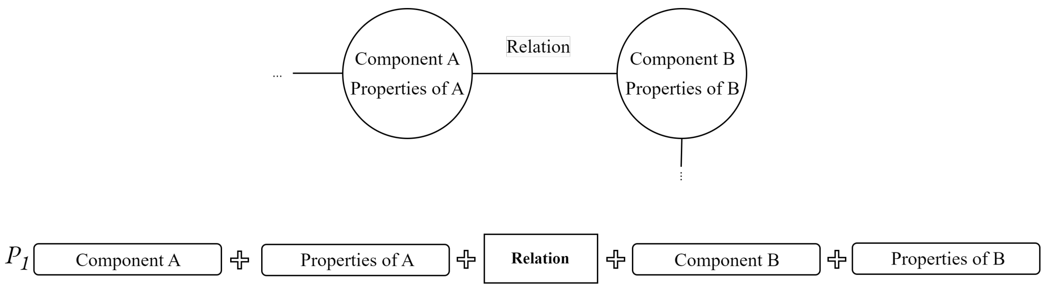

Figure 4. Example of transformation of information from graph representation to a pattern.

To retrieve information, a dataset of models was created. A connector was developed to extract embedded information from Simulink models, and these data were converted to text. This description was generated with a pattern containing the component names, types, relationships, and property information (see Figure 5).

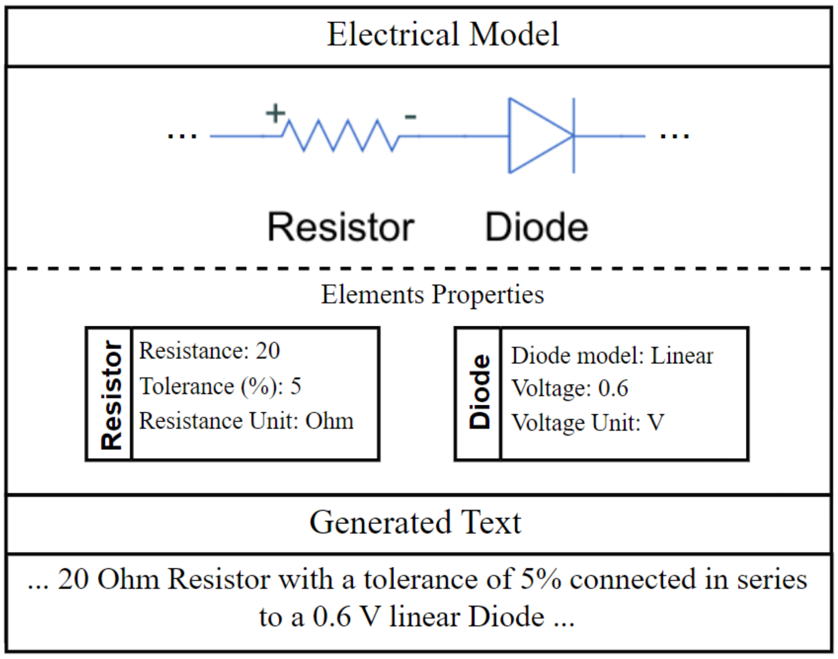

Figure 5. Example of the verbalisation process of an electrical model.

3.5. Search Algorithm

To carry out the retrieval process, an algorithm was designed and implemented based on pattern matching algorithms [31], vector representation techniques [32], and cosine similarity functions [33,34]. This algorithm calculates the similarity of the results from an input query to enable the retrieval of relevant physical models. In this section, a detailed explanation of the algorithm and its components is provided, along with a description of how the semantic reuse of physical models is achieved through pattern matching, vector representations, and cosine similarity.

1. **Pattern Matching:** in this step, the algorithm searches for the semantic pattern of the input query to be matched in the models. The model text and the query text are processed using NLP techniques, such as tokenisation and normalisation, to calculate the cosine similarity d between the resulting terms of the query t_q and model t_m. If the cosine similarity of t_q in t_m is consecutively higher than the value of 0.7 set in the algorithm, then the model is assigned a weight $w = 1000$. In this way, the models that match the pattern return a higher similarity than the others (see Figure 6).
   The threshold value of 0.7 was selected after an empirical experimentation process in which we tested different values ranging from 0.4 to 0.9. The selected threshold

consistently yielded the most accurate and relevant results while effectively balancing precision and recall. When setting the threshold at 0.7, the algorithm assigns a significant weight to highly relevant models that closely match the query's semantic pattern, thereby optimising the trade-off between capturing meaningful matches and avoiding excessive false positives. Similarly, the assigned weight $w = 1000$ for matching models was determined through experimentation, where it was found to provide a desirable emphasis on pattern matches while boosting the similarity scores. This weighted approach prioritises models that closely align with the query pattern, thereby refining the retrieval process. The combination of the threshold and weight parameters ensures that the algorithm generates meaningful and relevant results, facilitating the semantic reuse of physical models with improved precision and effectiveness.

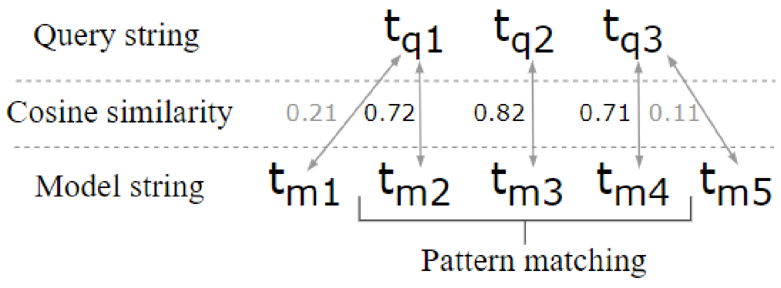

Figure 6. Example of the pattern matching calculation process.

2. **Term Similarity:** The last step of the algorithm consists of calculating the similarity between t_q and t_m. The cosine distance d is calculated for each element t_{qi} and t_{mj}, then multiplied by a weight function $h(x, y)$ provided by Equation 1, where x is the number of distance calculations with a value larger than a and less or equal to b and y is the number of distance calculations with a value smaller or equal to a. After conducting different tests and evaluations, the values of $a = 0.6$ and $b = 0.85$ were determined to offer the best precision in the algorithm. This weight function was implemented to take into account the relevance of terms in a query within a model.

$$h(x, y) = \begin{cases} \frac{1}{y}, & \text{for } a \geq d \\ \frac{1}{\sqrt{x}}, & \text{for } a < d \leq b \\ 1, & \text{for } b < d \leq 1 \end{cases} \tag{1}$$

Finally, the function to calculate the similarity of a model based on a query is presented in Equation (2). This function takes into account the cosine similarity between the query term (t_q) and model term (t_m) for each element in the query and model. The similarity score is then multiplied by a weight function $h(x, y)$ that considers the relevance of terms in the query within the model.

$$S_{model} = \sum_{i=0}^{n} \sum_{j=0}^{p} d(t_{qi}, t_{mj}) \cdot h(x, y) + w \tag{2}$$

4. Case Study: Searching Physical Models in MATLAB Simulink

In this study, we aim to address two key questions in the context of the presented case study: (1) Is it feasible to develop a semantic model to provide a search engine based on vector representations? and (2) Can this search engine deliver accurate results? To answer these questions, a case study using MATLAB Simulink was conducted to provide an advanced mechanism for indexing and retrieval of Simulink models. The experiment sought to evaluate the advantages of the proposed solution to retrieve physical models, and was designed as follows:

1. A dataset of Simulink models covering electronic models and physical models in the automotive and aerospace domains was defined. This dataset comprised 40 models.
2. A query dataset for evaluating the retrieval capabilities of the proposed solution was created. Queries were automatically designed by mixing the available terms in the physical models with other related terms that do not appear in the generated text of the model, such as the term circuit (see Table 1).
3. The indexing and retrieval process was executed. For each query defined in the previous step, we analysed the models retrieved via the implemented method, taking into account all matching term distances and patterns between the query and models.
4. The results were analysed and validated using the schema proposed in [35]. Specifically, the performance metrics used to evaluate the method were (1) precision (the proportion of retrieved information that is relevant); (2) recall (the proportion of relevant information that is retrieved); and (3) the F1 score, which is a combination of the first two metrics.

Table 1. List of queries executed to retrieve similar models.

Q	Query Description
Q_1	Circuits
Q_2	Models with a DC power supply of 200 V
Q_3	Models with hydraulic transfer functions
Q_4	Models with an integrator
Q_5	Electrical models using diodes
Q_6	Sum operations with integrators
Q_7	Resistor connected in series with a resistor
Q_8	Models with memory
Q_9	Circuits with supercapacitors
Q_{10}	Resistors connected in parallel with diodes
Q_{11}	Models with transfer functions
Q_{12}	Models with switches
Q_{13}	Models with voltages of at least 100 V
Q_{14}	Models with a 3-phase harmonic filter
Q_{15}	Models with SPICE resistors
Q_{16}	Physical models with torque
Q_{17}	Models with logical operations
Q_{18}	A transistor connected in parallel with a resistor
Q_{19}	Circuits with logical operations
Q_{20}	Battery of 12 V
Q_{21}	Models with sum operations
Q_{22}	Bang–bang controller
Q_{23}	AC voltage source of 100 V
Q_{24}	Models with functions
Q_{25}	Physical models with switches
Q_{26}	Models with a signal input
Q_{27}	A transistor in series with a resistor
Q_{28}	Aerospace models
Q_{29}	Capacitor connected with a voltage of 10 V
Q_{30}	Models with thermocouples

Analysis of Results

The results presented in this subsection are based on the levels of "goodness" (as outlined in Table 2) established in [36], a metric framework that has found successful application in similar retrieval works such as those in [25,26]. Our results show that, according to the metrics and the extracted performance metrics (see Table 3), all queries except four obtained the maximum precision value when retrieving physical models, which can be classified as excellent in terms of precision. Only one query performed below the excellent level for precision, which was because the implemented search algorithm did not take into account the semantic information about the different connection types between electronic components (parallel or series). In terms of recall, 58% of all executed queries achieved an excellent value, and all queries had values higher than what is considered good.

Table 2. Goodness levels for the precision and recall metrics [36].

Level of "Goodness"	Precision	Recall
Acceptable	≥20%	≥60%
Good	≥30%	≥70%
Excellent	≥50%	≥80%

Table 3. Precision, recall, and F1 metrics for each query.

	Precision	Recall	F1
Q_1	1.000	1.000	1.000
Q_2	1.000	1.000	1.000
Q_3	1.000	1.000	1.000
Q_4	1.000	0.941	0.970
Q_5	1.000	0.800	0.889
Q_6	1.000	0.842	0.914
Q_7	0.821	0.788	0.804
Q_8	1.000	1.000	1.000
Q_9	1.000	0.941	0.970
Q_{10}	0.375	0.462	0.423
Q_{11}	1.000	1.000	1.000
Q_{12}	1.000	1.000	1.000
Q_{13}	1.000	0.889	0.941
Q_{14}	1.000	1.000	1.000
Q_{15}	1.000	0.750	0.857
Q_{16}	1.000	1.000	1.000
Q_{17}	0.682	0.621	0.650
Q_{18}	1.000	0.933	0.965
Q_{19}	1.000	1.000	1.000
Q_{20}	1.000	0.571	0.727
Q_{21}	1.000	1.000	1.000
Q_{22}	1.000	1.000	1.000
Q_{23}	1.000	0.889	0.941
Q_{24}	1.000	1.000	1.000
Q_{25}	1.000	0.857	0.923
Q_{26}	1.000	0.889	0.941
Q_{27}	0.835	0.723	0.775
Q_{28}	1.000	0.873	0.932
Q_{29}	1.000	0.922	0.959
Q_{30}	1.000	1.000	1.000
Avg	0.925	0.865	0.884

When analysing the overall average metrics, it is important to highlight that the recall values are lower than the precision values. These findings indicate that while relevant models for a query are retrieved, there is some retrieval of irrelevant ones as well, which adversely impacts the value of the recall metric. This issue could be addressed by fine-tuning the weights and adjusting the penalty function to minimise the retrieval of irrelevant models and enhance overall performance.

The semantic search process conducted in the case study takes into account the content of the model alongside the generated description text. This contrasts with the approach presented in previous studies [25], where search was based on model metadata and a limited vocabulary was represented in an ontology.

The results of our case study show that the proposed approach represents a promising alternative that can assist engineers in reusing the content of physical models. By processing the distance between the terms embedded in the models using a trained model of word embedding, our method retrieves models with a higher level of precision and recall than the work presented in [25]. The proposed approach has the potential to save time and resources by allowing engineers to quickly identify and reuse relevant physical models.

The results for the precision, recall, and F1 metrics for each query, shown in Table 3, provide valuable insights that can help to address the two questions presented in this case study. (1) It is evident that the implemented semantic model search engine utilising pattern matching, vector representations, and cosine similarity is able to successfully retrieve relevant models for the majority of queries. (2) Moreover, the search engine demonstrates excellent accuracy, providing relevant and suitable results for the majority of the executed queries.

However, while the results are encouraging, we acknowledge that more work is needed to further improve the performance of the algorithm. In particular, we recommend conducting experiments in larger settings that incorporate real user needs, such as queries and large physical interconnected models. Testing the proposed approach with a wider range of use cases and user requirements will make it possible to better understand how it performs in real-world situations as well as to identify opportunities for further refinement.

5. Conclusions and Future Directions

In this work, we have emphasised the importance of representing and reusing models through the use of word embeddings as a means of improving the systems development process. To validate the presented approach, a case study was conducted representing and indexing a dataset of physical models from Simulink. A set of queries was designed to measure the performance of the retrieval system using a similarity algorithm that compares the distances of concepts between the model and queries. Through this case study, we have demonstrated the feasibility of the proposed approach in effectively supporting the two first stages (abstraction and selection) of a reuse strategy for a physical model.

In addition, this work provides insights into the performance of the similarity algorithm used in the process of reusing physical models. We found the algorithm to be effective in identifying relevant models for the queries as well as when the query contained terms that were not in the models; however, in certain cases irrelevant models were retrieved, leading to penalisation of the recall metric. As such, we suggest that future research focus on refining the algorithm to further improve its precision and recall values, for example, by readjusting the weights and penalty functions.

The use of word embeddings and similar approaches for promoting the reuse of physical models has the potential to revolutionise the engineering industry. By reducing the need for engineers to create a domain ontology to represent the semantics of the terminology, this methodology can speed up development times, improve efficiency, and allow for more complex systems to be developed. As such, we believe that continued research in this area is essential to unlocking the full potential of these techniques and maximising their impact on the engineering discipline.

However, as part of future work it is important to understand the specific purpose of a model and consider the viewpoints of stakeholders. To address potential confusion in terminology usage, frameworks such as LOTAR [37] have been developed to incorporate metadata that provide more precise definitions on the model's domain. This can help to clarify the intended purpose and perspective of the model, thereby reducing misinterpretations that may occur when diverse stakeholders hold varying viewpoints on the same concept.

Overall, we believe that our approach has the potential to make a significant impact in the field of physical modelling by providing a more efficient and effective way to reuse existing models. We hope that our work will inspire further research in this area and that it will ultimately lead to new tools and methods that make it easier for engineers to build and maintain high-quality physical models.

Author Contributions: Methodology, E.C.; software, E.C. and R.M; validation, E.C., R.M. and J.M.Á.-R.; formal analysis, E.C.; investigation, E.C., R.M. and J.M.Á.-R.; resources, E.C.; data curation, E.C.; writing—original draft preparation, E.C.; writing—review and editing, J.M.Á.-R.; visualisation, E.C.; supervision, J.M.Á.-R.; project administration, J.L.; funding acquisition, J.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially funded by H2020-ECSEL "Intelligent Reliability 4.0" (IREL 4.0; grant agreement: 876659), H2020-ECSEL Joint Undertaking (JU) "Arrowhead Tools for Engineering of Digitalisation Solutions" (grant agreement: 826452), and specific national programs (AEI/10.13039/501100011033).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The dataset of models used for conducting this case study is publicly available at https://github.com/edcisa/physicalModels. The dataset consists of 40 diverse and representative models created using Simulink. Each model captures a different aspect of physical systems encompassing various domains, functionalities, and complexities. Additionally, the list of 9620 terms used to train the model is available in the same repository, providing a comprehensive resource for further research and analysis.

Conflicts of Interest: Author Roy Mendieta was employed by the The Reuse Company. The remaining author declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Micouin, P. *Model-Based Systems Engineering: Fundamentals and Methods*; Control, Systems and Industrial Engineering Series; ISTE: London, UK, 2014.
2. Madni, A.; Madni, C.; Lucero, S. Leveraging Digital Twin Technology in Model-Based Systems Engineering. *Systems* **2019**, *7*, 7. [CrossRef]
3. Henderson, K.; Salado, A. Value and benefits of model-based systems engineering (MBSE): Evidence from the literature. *Syst. Eng.* **2021**, *24*, 51–66. [CrossRef]
4. Zeigler, B.P.; Mittal, S.; Traore, M.K. MBSE with/out Simulation: State of the Art and Way Forward. *Systems* **2018**, *6*, 40. [CrossRef]
5. Mili, H.; Mili, F.; Mili, A. Reusing software: Issues and research directions. *IEEE Trans. Softw. Eng.* **1995**, *21*, 528–562. [CrossRef]
6. Smolárová, M.; Návrat, P. Software reuse: Principles, patterns, prospects. *J. Comput. Inf. Technol.* **1997**, *5*, 33–49.
7. Kim, Y.; Stohr, E.A. Software reuse: Survey and research directions. *J. Manag. Inf. Syst.* **1998**, *14*, 113–147. [CrossRef]
8. Krueger, C.W. Software reuse. *ACM Comput. Surv. (CSUR)* **1992**, *24*, 131–183. [CrossRef]
9. Hogan, A.; Blomqvist, E.; Cochez, M.; d'Amato, C.; de Melo, G.; Gutiérrez, C.; Kirrane, S.; Labra Gayo, J.E.; Navigli, R.; Neumaier, S.; et al. *Knowledge Graphs*; Number 22 in Synthesis Lectures on Data, Semantics, and Knowledge; Springer: Berlin/Heidelberg, Germany, 2021. [CrossRef]
10. Madni, A.M.; Erwin, D.; Madni, C.C. Digital twin-enabled MBSE testbed for prototyping and evaluating aerospace systems: Lessons learned. In Proceedings of the 2021 IEEE Aerospace Conference (50100), Big Sky, MT, USA, 6–13 March 2021; pp. 1–8.
11. Wikidata. 2023. Available online: https://www.wikidata.org/wiki/Wikidata:WikidataCon_2023 (accessed on 10 January 2022).
12. Peres, R.S.; Jia, X.; Lee, J.; Sun, K.; Colombo, A.W.; Barata, J. Industrial artificial intelligence in industry 4.0-systematic review, challenges and outlook. *IEEE Access* **2020**, *8*, 220121–220139. [CrossRef]
13. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language models are few-shot learners. *arXiv* **2020**, arXiv:2005.14165.
14. Liu, R.; Fu, R.; Xu, K.; Shi, X.; Ren, X. A Review of Knowledge Graph-Based Reasoning Technology in the Operation of Power Systems. *Appl. Sci.* **2023**, *13*, 4357. [CrossRef]
15. Lehmann, J.; Isele, R.; Jakob, M.; Jentzsch, A.; Kontokostas, D.; Mendes, P.N.; Bizer, C. DBpedia—A large-scale, multilingual knowledge base extracted from Wikipedia. *Semant. Web* **2015**, *6*, 167–195. [CrossRef]
16. Pennington, J.; Socher, R.; Manning, C. Glove: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1532–1543.
17. TensorFlow. Word2Vec. Available online: https://www.tensorflow.org/text/tutorials/word2vec (accessed on 1 March 2023).
18. Stein, R.A.; Jaques, P.A.; Valiati, J.F. An analysis of hierarchical text classification using word embeddings. *Inf. Sci.* **2019**, *471*, 216–232. [CrossRef]
19. Tang, J.; Xue, Y.; Wang, Z.; Hu, S.; Gong, T.; Chen, Y.; Zhao, H.; Xiao, L. Bayesian estimation-based sentiment word embedding model for sentiment analysis. *CAAI Trans. Intell. Technol.* **2022**, *7*, 144–155. [CrossRef]

20. Satapathy, S.C.; Peer, P.; Tang, J.; Bhateja, V.; Ghosh, A. Machine Translation System Combination with Enhanced Alignments Using Word Embeddings. In *Intelligent Data Engineering and Analytics*; Smart Innovation, Systems and Technologies; Springer Singapore Pte. Limited: Singapore, 2022; Volume 266, pp. 19–29.
21. Berenguer, A.; Mazón, J.N.; Tomás, D. A Tabular Open Data Search Engine Based on Word Embeddings for Data Integration. In *New Trends in Database and Information Systems*; Chiusano, S., Cerquitelli, T., Wrembel, R., Nørvåg, K., Catania, B., Vargas-Solar, G., Zumpano, E., Eds.; Springer International Publishing: Cham, Switzerland, 2022; pp. 99–108.
22. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems 30, Proceedings of the 31st Annual Conference on Neural Information Processing Systems (NIPS 2017, Long Beach, CA, USA, 4–9 December 2017*; Neural Information Processing Systems Foundation, Inc. (NeurIPS): La Jolla, CA, USA, 2017.
23. Safavi, T.; Koutra, D. Relational world knowledge representation in contextual language models: A review. *arXiv* **2021**, arXiv:2104.05837.
24. Huang, B.; Zhang, S.; Huang, R.; Li, X.; Zhang, Y. An effective retrieval approach of 3D CAD models for macro process reuse. *Int. J. Adv. Manuf. Technol.* **2019**, *102*, 1067–1089. [CrossRef]
25. Cibrián, E.; Mendieta, R.; Rodríguez, J.M.Á.; Morillo, J.L. Towards the reuse of physical models within the development life-cycle: A case study of Simulink models. In Proceedings of the NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium, Budapest, Hungary, 25–29 April 2022; pp. 1–6.
26. Cibrian, E.; Alvarez-Rodriguez, J.M.; Mendieta, R.; Llorens, J. Discovering traces between textual requirements and logical models in the functional design of Printed Circuit Boards. In Proceedings of the 2022 IEEE 5th International Conference on Industrial Cyber-Physical Systems (ICPS), Coventry, UK, 24–26 May 2022; pp. 1–6.
27. *ISO 10303-243:2021*; Industrial Automation Systems and Integration—Product Data Representation and Exchange. International Organization for Standardization: Geneva, Switzerland, 2021.
28. Rodríguez, J.M.Á.; Mendieta, R.; de la Vara, J.L.; Fraga, A.; Morillo, J.L. Enabling System Artefact Exchange and Selection through a Linked Data Layer. *J. Univers. Comput. Sci.* **2018**, *24*, 1536–1560.
29. NLPL Word Embeddings Repository. Available online: http://vectors.nlpl.eu/repository/ (accessed on 20 June 2022).
30. MathWorks. Llamar a MATLAB desde NET-MATLAB & Simulink-MathWorks España. 2023. Available online: https://es.mathworks.com/help/matlab/call-matlab-from-net.html (accessed on 28 June 2023).
31. Ouyang, W.; Tombari, F.; Mattoccia, S.; Di Stefano, L.; Cham, W.K. Performance evaluation of full search equivalent pattern matching algorithms. *IEEE Trans. Pattern Anal. Mach. Intell.* **2011**, *34*, 127–143. [CrossRef] [PubMed]
32. Roy, D.; Ganguly, D.; Mitra, M.; Jones, G.J. Representing documents and queries as sets of word embedded vectors for information retrieval. *arXiv* **2016**, arXiv:1606.07869.
33. Jatnika, D.; Bijaksana, M.A.; Suryani, A.A. Word2vec model analysis for semantic similarities in english words. *Procedia Comput. Sci.* **2019**, *157*, 160–167. [CrossRef]
34. Lahitani, A.R.; Permanasari, A.E.; Setiawan, N.A. Cosine similarity to determine similarity measure: Study case in online essay assessment. In Proceedings of the 2016 4th International Conference on Cyber and IT Service Management, Bandung, Indonesia, 26–27 April 2016; pp. 1–6.
35. Juristo, N.; Moreno, A.M. *Basics of Software Engineering Experimentation*; Springer Science & Business Media: New York, NY, USA, 2013.
36. Hayes, J.H.; Dekhtyar, A.; Sundaram, S.K. Improving after-the-fact tracing and mapping: Supporting software quality predictions. *IEEE Softw.* **2005**, *22*, 30–37. [CrossRef]
37. Coïc, C.; Williams, M.; Mendo, J.C.; Alvarez-Rodriguez, J.M.; Richardson, M.K. Linking Design Requirements to FMUs to create LOTAR compliant mBSE models. In Proceedings of the 15th International Modelica Conference, Aachen, Germany, 9–11 October 2023.