Graph based Smart Neural Recommendation Mechanism (SNeRM) for system modelling

Basil Cardoz
SDT-PLO
Siemens Technology and Services Private Limited, India
Bangalore, India
https://orcid.org/0000-0002-6252-9546

Samir Mishra
Department of Computer Science
International Institute of Information Technology
Bangalore, India
msamir.dls@gmail.com

Ganesh S
SDT-PLO
Siemens Technology and Services Private Limited, India
Bangalore, India
s.ganesh@siemens.com

***Abstract* - Creating a representation of real systems to evaluate its behavior can become very complex once multiple sub-systems are created that interact with one another. This can become especially challenging for inexperienced users who are trying to test out different concepts or validate with existing 3D real time systems. These challenges are mainly faced due to uncertainty on how to model or represent certain sub-systems, interconnecting different concepts and validation of results obtained from simulations.**

This is where Smart Neural Recommendation Mechanism (SNeRM) with in-domain knowledge can aid users while designing these systems. These recommendation systems can be embedded with knowledge by training it on models created in the past by experts and build an understanding of the context with which the models were developed. These trained systems can assist other users with further development by providing recommendations based on the current intent of the user in an implicit manner.

In our current work, we have investigated the ability of contextually rich structured graphs to understand the different components within each circuit or system as well as the functional relationship between the components. These graphs are then broken down into smaller subgraphs to easily match with the circuit the user intends to create through a technique called Component Neuro Sub-Graph Matching

Keywords—Graph Neural Network, Knowledge Graph, Recommendation Engine, System Modelling

I. INTRODUCTION

System modelling requires expertise to understand systems and sub-systems and requires the skills to design using fundamental building blocks. For this, a theoretical as well as practical knowledge of the system is essential. In the aerospace industry, New Product Ingestion (NPI) for components such as aircraft engines requires multiple phases of development like Conceptual Design (CD), Preliminary Design (PD) and Detailed Design (DD). Architectures of the various subsystems such as cooling, combustion, mechanical components, and control systems are evolved and finalized in the initial phases of the development. Careful system design is essential to prevent the squandering of predevelopment resources and time and to guarantee safety during system testing. Hence, for aircraft engine related NPI development typically takes around 4-5 months for architecture to pass conceptual design reviews. A significant amount of time is spent in adhering to design practices (DP) and design guidelines while designing such critical components. Additional time is spent in exploring novel architectures for 1D design modelling. Therefore, there is a need for an intelligent mechanism that can assist design engineers during the system modelling phase which can validate the designs based on past domain knowledge. Such a system would need to have an extensive knowledge of functional relationship between components and sub-component along with the right knowledge on the overall objective considering the individual component characteristics.

Earlier researchers address the above problem of knowledge modelling in conceptual design environment through building of knowledge component engine/databases [1] and employ bilinear function to capture the co-occurrence patterns of related items through Memory Augmented Graph Neural networks [2]

In this paper, we are proposing Smart Neural Recommendation Mechanism (SNeRM), a novel deep learning-based architecture inspired from [3] that can automatically recommend solutions in 1D system modelling to a design engineer in an implicit manner. The overall architecture utilizes contextually rich graphs that are created from domain specific models and capture the functionality of different components, their inter-connection, and the overall modelling intent. Using the Component Neuro Sub-graph matching, we dissect the 1D connected modules into sub-modules (sub graph) in an automated k-hop traversal and further, utilize the contrastively learnt graph embeddings to recommend the most probable network/modelling component.

The overall objective of this study is to show the overall effectiveness of graph-based recommendations system to assist users during the system modelling process of component designs.

II. ANALYSIS

A. Graph database generation for recommendations

The recommendation engine shown in this study is a type of content-based recommendation system where a similarity function between query and database 1D models is used to recommend more feasible next steps. In real time engineering systems (e.g : Cooling Systems in Aero Engines Gas Turbines, PCB design of electrical component, Refrigeration in air condition system), during the conceptual design phase, system modelling typically involves building/connecting multiple sub components (like valves, bearing, couplers) through 1D network modelling tools (e.g: AMESim [4] , Dymola [5]). An example of 1D modelling networks diagram (Aircraft Refrigeration System) is shown in Fig. 1 where for a given components like compressor, heat exchanger, there are specific domain rules embedded in the forms of the type of connection with the adjacent components, specific ports to be connected etc. In the current recommendation system, SNeRM model learns this domain specific relationship.

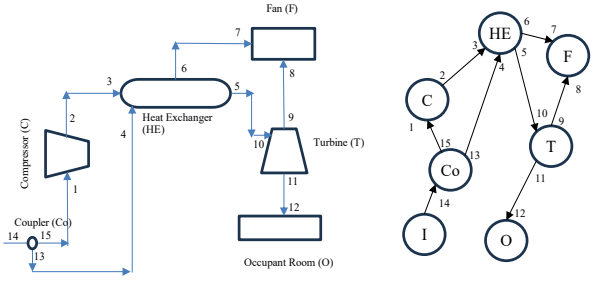

Figure 1: 1D modelling network diagram for Aircraft Refrigeration System with Graph representation

between different components and make adaptive recommendations of the possible next steps understanding the current user state during modelling. To achieve the above objective, the conversion of the 1D models into domain graphs where each graph represents a particular design module is adopted. Particularly, the database containing multiple AMESim files corresponding to 1D designs of sub-systems across varied domains (Automobile, Aerospace, Electrical) are taken into consideration. The 1D design files graph database involves the translation of Python design macro scripts to structured graphs with the components/sub-components representing the design nodes and the connections between them representing the predicates. The type of components, their input/output ports and descriptions are embedded as features of the nodes (component embedding vocabulary) which will be detailed in Section B.

B. Recommendation Methodology

In order to build a component embedding vocabulary, each of the unique components that were used to create the different circuits are assigned a particular index number as well as the number of ports for each component were identified. To make the embedding more contextually rich, the component descriptions are obtained by providing a suitable prompt to a Large Language Model. To convert the description into a vector embedding, a sentence transformer model was used. Since the description of the components were not too large, a reduction in the description embedding size was performed to reduce the features needed to be learnt by the order embedding model. Finally, the 3 different features sets are concatenated to form a component embedding layer.

Using the features extracted from each AMESim circuit, a contextually rich graph of the circuit was created by interconnecting the relevant nodes with features from the component embedding vocabulary. These graphs were then used to train the Order embedding model (detailed in section 2.3) which creates an embedding to classify if a particular query graph is a subgraph of the target graph.

Overall methodology followed to create the Smart Neural Recommendation Mechanism (SNeRM) has been shown in Fig. 2. During the system modelling process, the SNeRM obtains the components added by the user in the design space which it treats as the Query Graph. Using the components present, additional feasible graphs are created which take into account a number of factors and are treated as Target graphs. These query and target graphs are then converted into the order embedding space to identify which of the feasible graphs are the best suited target graph based on the embedding value. This is then used to suggest the potential recommendations to the user.

C. Order Embedding Model

In this paper, we use the concept of order embedding based on (3), a novel technique that plays a pivotal role in searching for isomorphic subgraphs over target graphs, since the subgraph relationships induce a partial ordering over subgraphs introducing geometrical constraints over graphical structures, allowing us to use order embedding.

Order embedding is a special type of monotonic functions which transforms features into continuous vector representation preserving both inherent ordering and order reflection, enabling model to understand user intent and recommend accordingly.

Order embedding captures subgraph relationship, consider graphs G_1 and G_2, $G_1$is subgraph of G_2 then nodes of G_1 and G_2 must satisfy the constraint, embedding for node in query graph must be less than equal corresponding node embedding of target graph. This property allows the node embedding of node in G_1 will be below right of the embedding of corresponding node in G_2. Once order Embeddings are calculated, max-margin loss function is used to capture the subgraph relations, enhancing the recommendations.

D. Subgraph Prediction function

For any given node embedding in the target graph (z_u) where $u \in G_T$ and node embedding in the query graph (z_q) where $q \in G_Q$, the subgraph prediction function decides whether if u has k-hop neighbourhood that is a isomorphic subgraph of q's k-hop neighbourhood. The prediction is made only based on the embedding, if q is a subgraph of u, then the order embedding of z_q must be to the lower-left to the embedding of z_u.

$$z_q[i] \leq z_u[i] \ \forall_{i=1}^{D} \quad if \ G_q \subseteq G_u \tag{1}$$

where D is the embedding dimension. Hence, we train the GNN (Graphical Neural Network) that learns the order embeddings based on the max margin loss:

$$\mathcal{L}(z_q, z_u) = \sum_{(z_q, z_u) \in P} E(z_q, z_u) + \sum_{(z_q, z_u) \in N} max\{0, \alpha - E(z_q, z_u)\} \tag{2}$$

$$E(z_q, z_u) = || \, max\{0, z_q - z_u\} ||_2^2 \tag{3}$$

Here P denotes the set of positive examples in minibatch where the neighborhood of q is a subgraph of neighborhood of u, and N denotes the set of negative examples.

E. Provide recommendation to user

Based on the subgraph prediction function mentioned in the previous subsection, the graph of the components present within the 1D modelling design space is considered as the query graph with an embedding z_q. Multiple feasible components can be added to the current design space to produce multiple target graphs with each graph having a particular embedding z_u. Any of the multiple graphs produced can be considered as a valid recommendation if the query graph node embedding lie in the lower-left of the target graph node embedding as shown in Fig. 3.

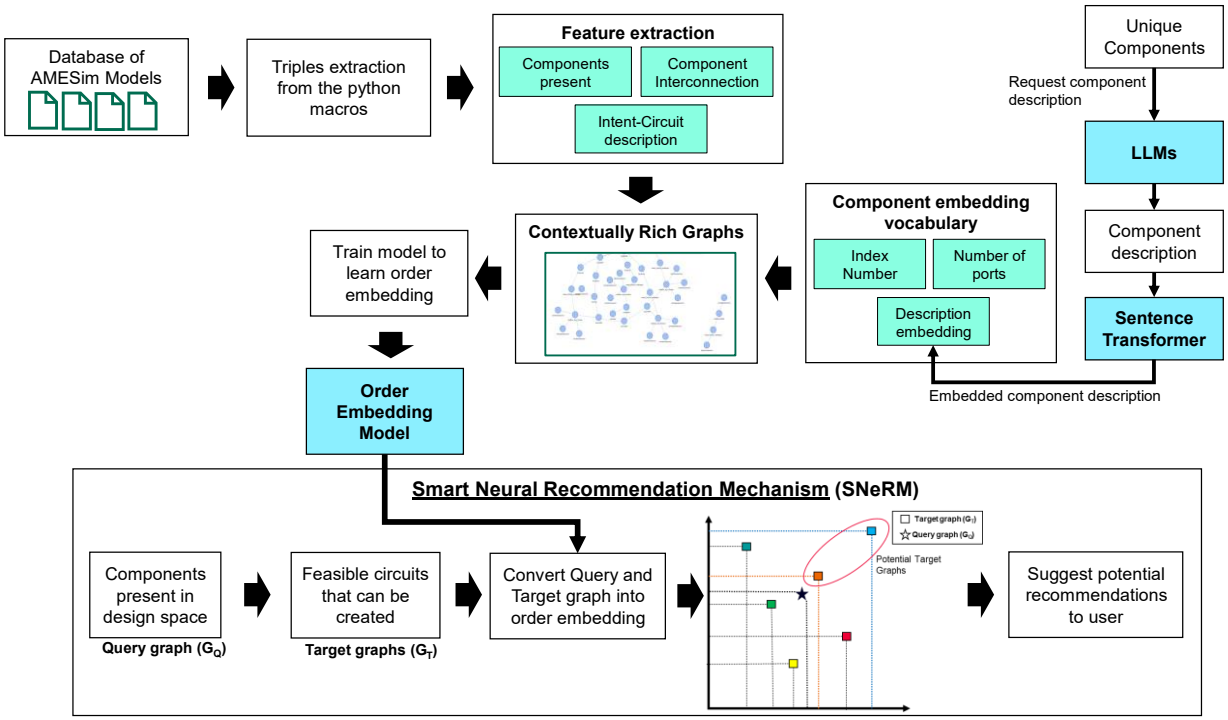

Figure 2: Methodology

III. Results and Discussion

A. Recommendation mechanism

The graph-based recommendation neural network is trained on 1D modelling dataset consisting of around 40 AMESim circuits. The features such as components present, their interconnection as well as the circuit description were extracted from the python scripts associated with these circuits. These circuit features were then used to construct graphs with interconnected nodes consisting of nodes with embeddings obtained from the component embedding vocabulary.

The embedding vocabulary consists of 104 unique components which were assigned unique index values as well as number of ports for each of the components were assigned based on values observed in the scripts. The description embedding was created by firstly obtaining the description of each components through use of prompts on ChatGPT using LangChain to automate the process. The description were then passed through the all-MiniLM-L6-v2 sentence transformer since is it faster than other available models while still offering good quality. The model embeds text to a 384-dimensional dense vector space however, further dimension reduction was performed to 32-dimensional vector space by using PCA (Principal Component Analysis) Technique. This gave an overall node-level embedding size of 36 consisting of index number, number of ports and the description embedding.

The graphs constructed were then used to train the Order Embedding model. Learning rate was set to 10^{-5} and embedding size of 64 was used.

B. Recommendation Validation

To test the accuracy of the recommendations to the users, 29 different scenarios were considered. From the already existing circuits, a three-component circuit was obtained and assumed to be the initial components present in the design space. This circuit in the graphical form is the query graph (G_Q) and 4 corresponding target graphs were created. Target graph G_{T1} and G_{T2} were the actual circuit graph with additional 1 and 2 correct nodes respectively. The target graphs G_{T3} and G_{T4} were incorrect circuit graphs with 1 and 2 randomly chosen incorrect nodes respectively. The target graph G_{T5} contains one correct node and one incorrect node. An example for the graphs that were generated are shown in Fig. 4.

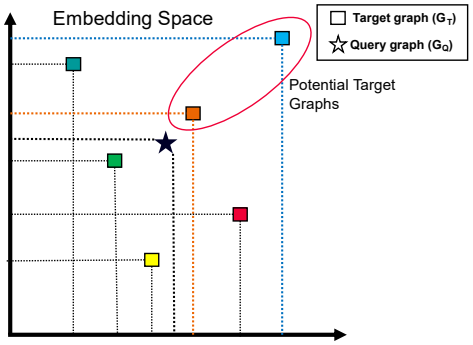

Figure 3: Recommendation based on order embeddings.

Each node in the query was compared with the order embedding for nodes the target graphs based on the subgraph prediction function (3). This provides a matrix with rows as query nodes and columns as target graph nodes also referred to as the alignment matrix. The alignment matrix for the target graphs G_{T2} and G_{T4} are shown in Fig. 5. Matrix elements with value of zero means that for that particular anchor node combination, the query is a subgraph of the target graph based on the values learnt by the GNN. Whereas a non-zero value indicates some misalignment in the order embedding space. To identify if the order embedding was able to correctly

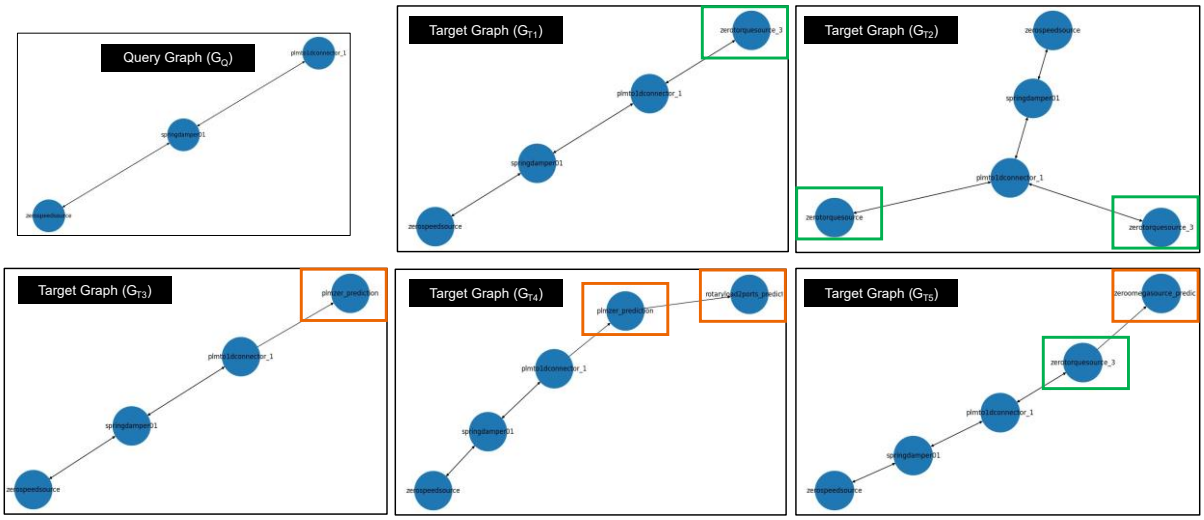

Figure 4: Query graph (G_Q) and Target graphs (G_{T1}, G_{T2}, G_{T3}, G_{T4}, G_{T5})

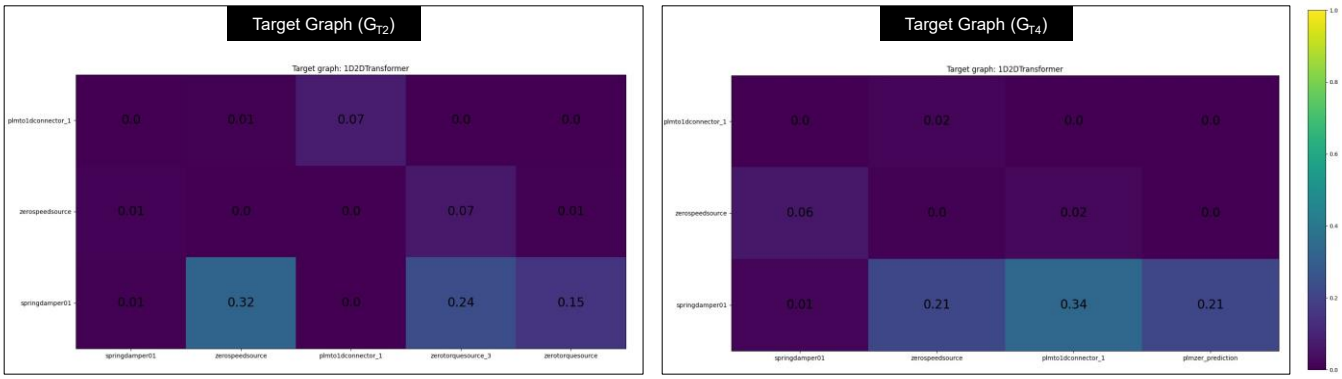

Figure 5: Alignment matrix for Two-level node prediction Target Graph T_{G2} and T_{G2}

	One Level Node Prediction		Two Level Node Prediction			Difference	Rank of TG2
	TG1	TG3	TG2	TG4	TG5	Δ(TG4,TG2)	
QG1	0.001	0.015	0.0000	0.0214	0.0077	0.021	1
QG2	0.000	0.037	0.0000	0.0010	0.0016	0.001	1
QG3	0.001	0.002	0.0000	0.0054	0.0000	0.005	1
QG4	0.000	0.000	0.0000	0.0168	0.0134	0.017	1
QG5	0.000	0.004	0.0022	0.0015	0.0013	-0.001	3
QG6	0.041	0.010	0.0000	0.0010	0.0019	0.001	1
QG7	0.019	0.006	0.0001	0.0006	0.0050	0.001	1
QG8	0.003	0.002	0.0009	0.0023	0.0020	0.001	1
QG9	0.008	0.166	0.0307	0.1710	0.0242	0.140	2
QG10	0.000	0.007	0.0000	0.0006	0.0000	0.001	1
QG11	0.055	0.028	0.0943	0.0904	0.0142	-0.004	3
QG12	0.002	0.001	0.0001	0.0054	0.0057	0.005	1
QG13	0.132	0.069	0.1702	0.1482	0.0800	-0.022	3
QG14	0.002	0.000	0.0062	0.0249	0.0036	0.019	2
QG15	0.040	0.173	0.0154	0.2008	0.0569	0.185	1
QG16	0.000	0.000	0.0000	0.0274	0.0015	0.027	1
QG17	0.000	0.001	0.0000	0.0022	0.0007	0.002	1
QG18	0.016	0.018	0.0671	0.0424	0.2094	-0.025	2
QG19	0.003	0.012	0.0001	0.0010	0.0002	0.001	1
QG20	0.000	0.001	0.0000	0.0021	0.0000	0.002	1
QG21	0.011	0.008	0.0000	0.0011	0.0000	0.001	1
QG22	0.000	0.001	0.0000	0.0064	0.0016	0.006	1
QG23	0.000	0.012	0.0000	0.0232	0.0049	0.023	1
QG24	0.002	0.000	0.0001	0.0006	0.0000	0.000	2
QG25	0.002	0.000	0.0000	0.0017	0.0010	0.002	1
QG26	0.001	0.018	0.0000	0.0092	0.0000	0.009	1
QG27	0.000	0.001	0.0077	0.0012	0.0181	-0.006	2
QG28	0.231	0.025	0.0771	0.0595	0.0036	-0.018	3
QG29	0.028	0.015	0.0010	0.0017	0.0097	0.001	1

Figure 6: Mode of the aligment matrix for target graphs with query graphs.

differentiate between the correct and incorrect target graph predictions, the mode of the alignment matrix was used as it refers to the most frequent value.

C. *Results*

The mode of the alignment matrix for each query and corresponding target graphs along with the difference in the mode values for only the two-level node prediction target graphs G_{T2} and G_{T4} in shown in Fig. 6. The order embedding model was correctly able to differentiate between the correct and incorrect prediction with an accuracy of 79.31% for the two-level node prediction whereas the accuracy for one-level node prediction was around 55%. In terms of Recommendation Metric, Mean Reciprocal Rank (MRR), the current model delivers >82.1%. This shows that the model can provide the user with a correct recommendation of the potential next components that he would like to add to the current circuit being designed without any user intervention.

IV. Conclusion

In this paper, we have demonstrated the feasibility of design assistance/recommendation engine using sub-graph matching techniques on top of existing 1D design models. Further by testing on real-time cases, we have been able to accomplish higher accuracy/MRR scores. Some unique novel outcomes from this effort are:

- Knowledge fusion of component description and their functional relationships which can be utilized for objective based learning.
- Deliver on an inductive graph intelligence architecture which can used in domain specific transfer learning.

- Show the value of recommendations (that are mostly prevalent in information retrieval world) for real time mechanical/electrical system design engineering.

With these technique, it should help in reducing manual efforts and leveraging the database knowledge for 1D System modelling.

V. Acknowledgement

We would like to extend our heartfelt appreciation and gratitude to our fellow colleagues who have generously dedicated their time and effort to assist from the conceptualization to the writing and proofreading of this paper. Their invaluable contributions and constructive feedback have greatly enhanced the quality and clarity of the content. Their support and collaboration have been instrumental in making this research endeavour successful. Special thanks to Vinay Ramnath and Krishna Chaitanya for suggesting and allowing us to publish our work in this conference.

VI. References

1. (FENG Haocheng, LUO Mingqiang, LIU Hu, WU Zhe, 2011) - A Knowledge-based and Extensible Aircraft Conceptual Design Environment. School of Aeronautic Science and Engineering, Beihang University, Beijing 100191, China.
2. (Ma, C., Ma, L., Zhang, Y., Sun, J., Liu, X., & Coates, M., 2020) - Memory Augmented Graph Neural Networks for Sequential Recommendation. Proceedings of the AAAI Conference on Artificial Intelligence, 34(04), 5045-5052.
3. (Ying, R., Lou, Z., You, J., Wen, C., Canedo, A., Leskovec, J., 2020) - Neural Subgraph Matching.
4. AMESim. (2023). AMESim Software. Version 5.0.0. Siemens Industry Software Inc.
5. M. Dempsey, "Dymola for Multi-Engineering Modelling and Simulation," *2006 IEEE Vehicle Power and Propulsion Conference*, Windsor, UK, 2006, pp. 1-6, doi: 10.1109/VPPC.2006.364294.
6. (Matthias Fey, Jan Eric Lenssen)- Fast Graph Representation Learning with PyTorch Geometric, Published as a workshop paper at ICLR 2019