

IDETC-CIE 2022/DAC-88049

MATERIAL PREDICTION FOR DESIGN AUTOMATION USING GRAPH REPRESENTATION LEARNING

Shijie Bian
Department of Mathematics
University of California
Los Angeles, CA 90095, USA

Daniele Grandi, Kaveh Hassani
Autodesk Research
Autodesk Inc.
San Rafael, CA 94903, USA

**Elliot Sadler, Bodia Borijin,
Axel Fernandes**
Autonomy Research Center for STEAHM
California State University Northridge
Northridge, CA 91324, USA

Andrew Wang
Portola High School
Irvine, CA 92618, USA

Thomas Lu, Richard Otis
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109, USA

Nhut Ho, Bingbing Li*
Autonomy Research Center for STEAHM
California State University Northridge
Northridge, CA 91324, USA
*bingbing.li@csun.edu

ABSTRACT

Successful material selection is critical in designing and manufacturing products for design automation. Designers leverage their knowledge and experience to create high-quality designs by selecting the most appropriate materials through performance, manufacturability, and sustainability evaluation. Intelligent tools can help designers with varying expertise by providing recommendations learned from prior designs. To enable this, we introduce a graph representation learning framework that supports the material prediction of bodies in assemblies. We formulate the material selection task as a node-level prediction task over the assembly graph representation of CAD models and tackle it using Graph Neural Networks (GNNs). Evaluations over three experimental protocols performed on the Fusion 360 Gallery dataset indicate the feasibility of our approach, achieving a 0.75 top-3 micro- F_1 score. The proposed framework can scale to large datasets and incorporate designers' knowledge into the learning process. These capabilities allow the framework to serve as a recommendation system for design automa-

tion and a baseline for future work, narrowing the gap between human designers and intelligent design agents.

1 INTRODUCTION

Material selection is a critical step in designing and manufacturing a product. Materials are integral to a product's success, safety, and efficiency [1]. Engineers and designers spend a considerable amount of time evaluating trade-offs between materials, such as cost, performance, manufacturability, and sustainability [2]. Each material needs to support the product's primary function while also withstanding uncertain environmental conditions that the product undergoes. Selecting an improper material can lead to extra expense, low structural integrity, and a shortened product life cycle [3]. Designers mainly rely on their experience and existing guidelines to evaluate trade-offs between different materials. Designing intelligent tools based on knowledge graphs and large design data could help facilitate this process. In product design, efforts to consolidate design knowledge in knowledge graphs have resulted in robust graph representations of domain-specific semantic relationships [4, 5]. Engineer-

*Address all correspondence to this author.

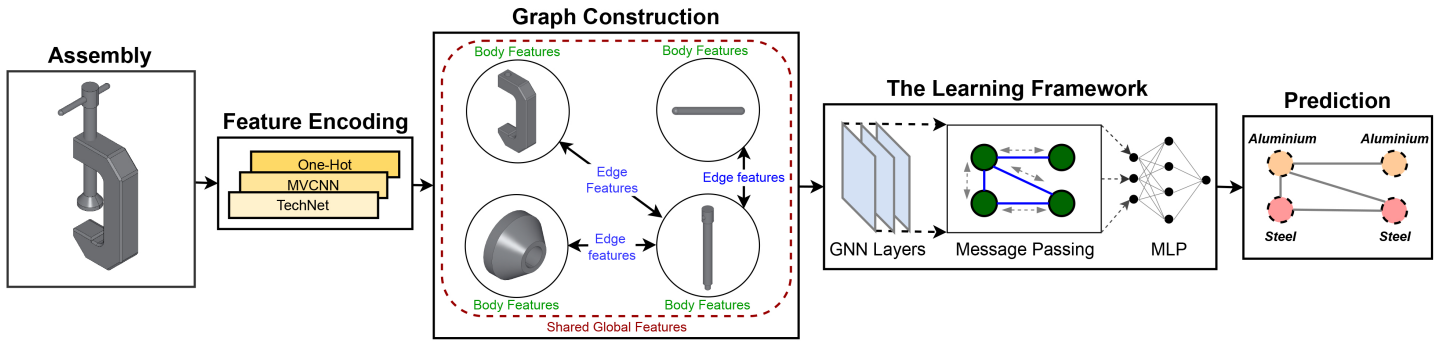


FIGURE 1. PROPOSED FRAMEWORK FOR PREDICTING MATERIALS OF ASSEMBLY BODIES USING GRAPH REPRESENTATION LEARNING.

ing design knowledge graphs are useful for concept generation and evaluation [6]. However, there is an opportunity to expand on prior work towards later phases of the design process by extracting design knowledge from computer-aided design (CAD) models to complement existing semantic networks.

CAD tools are used to create 2D and 3D models of the products and document various aspects of the design, including the geometries, dimensions, tolerances, degrees of freedom, and relative motions of parts, as well as the material of each component in the assembly [7]. The material information stored in CAD tools is typically used for simulation or rendering workflows. Both activities help designers assess and visualize trade-offs between different materials and identify the best materials for the design.

In recent years, several large datasets of curated CAD models have been released that support machine learning methods for various data-driven design applications [8, 9, 10]. Simultaneously, graphs have been leveraged to represent design data and to capture various relationships, including joint relationships between parts in assemblies [11, 12], semantic relationships between engineering and design concepts [4], and geometric relationships between faces of BREPs [13, 14].

In product design, efforts to consolidate design knowledge in knowledge graphs have resulted in robust graph representations of domain-specific semantic relationships. Engineering design knowledge graphs have been shown to be useful for concept generation and evaluation [5]. Leveraging the expressive power of graphs in capturing multi-modal information, we propose a framework (Figure 1) in which material prediction is posed as a node prediction task and is tackled through learning representations using graph neural networks (GNNs). Motivated by the importance of material selection to support design automation, this unified framework can help designers select appropriate materials by providing part-level material suggestions given a product assembly.

The method is validated through three experiments simulating different applications using the Fusion 360 Gallery Assem-

bly Dataset [11]. Also, through an ablation study of features and GNN layers, additional insights are provided into factors that influenced the performance, which may serve as a preliminary discussion and a baseline for future studies.

To summarize, our work makes the following contributions to the areas of design automation and engineering:

1. We study the material selection task for design automation and devise a systematic procedure to represent CAD models as graphs by extracting and encoding multi-modal features.
2. We leverage GNNs to learn expressive representations from existing CAD assemblies and use them for predicting materials on new assemblies.
3. We provide insights into the importance of features commonly found in CAD with respect to material prediction and evaluate the framework within three design scenarios.

To enable further research and reproducibility, we share the code for feature extraction, training, and experiments ¹.

2 RELATED WORK

We review the prior work related to material selection, graph representation learning in design automation, and graph neural networks. Reflecting on these past works, we provide deeper insights into the motivation and vision of our work.

2.1 Material Selection Design

In product design, material selection can be broken down into a five-step procedure: (1) establishing design requirements, (2) screening materials, (3) ranking materials, (4) researching material candidates, and (5) applying constraints to the selection process [1]. Many properties influence material selection, and the needs of a project determine the most important factors to be considered. This step is heavily reliant on performance indices and material property charts (Ashby charts) [1] which plot one

¹<https://github.com/danielegrandi-adsk/material-gnn>
Copyright © 2022 by ASME

material property against another, resulting in clusters of material classes. Then, the material search can be further narrowed down by setting axis limits [15]. In practice, tools such as Granta CES Edupack software can be used to compare over 4,000 materials based on user-input design requirements and constraints [16]. Product design aims to meet customer needs of technical requirements [17]. The material selection factors become more ambiguous because of the wide range of consumer products and multiple ways of designing each product. Material aspects such as quality, cost, and function must be considered in the design of user products because they interact and contribute to the final design [18]. Another important approach focuses on sustainability as a consideration of the economic, environmental, and societal impacts during the whole life-cycle from cradle to stage [19]. When working with complex assemblies, the material selection process is further complicated, as the material of individual parts affects the assembly integration process [20]. Thus, assembly data should be considered during the material selection process.

2.2 Graph Neural Networks (GNNs)

GNNs are a class of deep neural networks to perform inference on data described by graphs in a non-Euclidean space. GNNs learn node representations over order-invariant and variable-size data, structured as graphs, through an iterative process of transferring, transforming, and aggregating the representations from topological neighbors. The learned representations are then summarized into a graph-level representation [21, 22, 23, 24, 25, 26]. Utilizing the process of representation learning, GNNs can perform tasks such as graph, node, and edge classifications, and are applied to real-world applications such as point cloud segmentation [27], robot designs [28], physical simulations [29, 30], quantum chemistry [31], material design [32], semantic role labeling [33], and product relationship predictions [34].

2.3 Graph Representation of CAD Models

The design process as a whole is iterative and generates large amounts of data that can be organized and parsed for additional information that may be used to improve the design [35]. This information, collected from all aspects of the product life-cycle, is multi-modal and can be in the form of semantic names, customer requirements, 3D geometry, material properties, manufacturing tolerances, cost information, etc. This data may then be used to modify the design itself, enabling some automation of the design process by learning from prior examples [36, 37]. Prior work looked at organizing and learning from design knowledge acquired from sources such as taxonomy-based design repositories [5, 38], product teardowns [39], patent data [4], and geometry-based design repositories [40, 41, 42]. When working with geometric data, graphs have been leveraged to represent CAD models with goals ranging from representation of complex relations to solving design problems [43, 44, 45, 13, 14]. The current work

expands the representation of design into multi-modal representation, capturing geometric and semantic information in addition to the hierarchical structure of parts in assemblies.

3 METHODOLOGY

This section provides an overview of the framework, the process and motivation for data selection, and the detailed methodology for predicting materials of assembly bodies using graph representation learning.

As illustrated in Figure 1, the proposed framework consists of three main modules: (1) the **feature encoding** module extracts useful features from assembly files (such as semantic and physical properties of bodies, assembly connections between each body, and global features shared across bodies) and encodes them into embedding vectors; (2) the **graph construction** module structures assemblies, along with their bodies and the encoded features, into assembly graphs; (3) the **learning framework** trains GNNs upon the graph representations to learn robust embeddings that are used for predicting materials.

3.1 Dataset

The proposed framework is developed to work with the publicly available Fusion 360 Gallery Assembly Dataset [11]. As illustrated in Figure 2, each *assembly* in the dataset contains several *bodies* with a set of design features. Users organize *bodies* by placing them in a hierarchy of *occurrences*, which are the building blocks that make up *assemblies*. The dataset contains CAD data from 154,468 bodies, grouped into 8,251 assemblies, from different industries and with various levels of detail. It was chosen because of its size, diversity of designs and features, ease of processing, and presence of per-body material labels. We use the train/test split provided with the dataset for all experiments. For details on features and the encoding methods refer to Table 4 in Appendix A.1. We applied several pre-processing steps to the dataset as follows.

Data Filtering Many bodies in the dataset are labeled with the default material assigned at the time of creation with the CAD tool. Suppose all the bodies of an assembly are made of the default material, then it might be assumed that the user did not perform intentional material selection, rendering the data unusable for learning. Therefore, a filtering step was taken to remove the assemblies that have all bodies labeled with the default material and default appearance. 5388 out of 8251 assemblies were left after this step.

Material ID Transformation Each body in the dataset has two material labels: a physical material label that defines the mechanical and physical properties of the body, and an appearance label used for rendering. While each label serves a different purpose, it

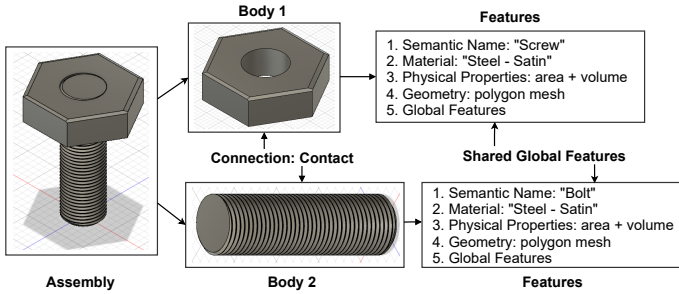


FIGURE 2. AN EXAMPLE OF AN ASSEMBLY WITH TWO BODIES AND THEIR FEATURES.

can be assumed that the designer intends each body to be defined by only one material. Therefore, for bodies with default physical material but non-default appearance labels, the non-default appearance was used as the ground truth. This was done to improve the quality and number of ground-truth labels.

Material Grouping The user-defined body materials are significantly diverse, skewed, and sparse in distribution, as shown in Table 5 in Appendix A.2. Considering that the materials with infrequent occurrences are unlikely to appear in future test cases and may not generalize well to other design use cases, only the top 20 material categories were preserved. Other materials were grouped into a single category.

3.2 Feature Encoding

This section describes the definition and encoding of valuable features from the dataset. These encoded features are later used to generate relational assembly graphs in Section 3.3 to feed into the learning architecture.

Semantic Names Each body has a `body name` and an `occurrence name`. Users can manually set these semantic names or leave them as default. TechNet [4], a semantic network with support for processing technology-focused texts, was used to pre-process and vectorize the semantic names. The cleaned semantic names were passed to TechNet to extract and retrieve keywords and corresponding 600-dimensional embeddings of keywords. The unmatched names were assigned a feature vector imputed according to the distribution of the TechNet database, while bodies with the default name were assigned a vector of zeros. The average embedding of all the keywords was assigned to the semantic name for semantic names with multiple keywords. The process is shown in Figure 3.

Physical Properties Each body has three physical properties: `body area` (in square meters), `body volume` (in cubic meters), and `body center of mass` (represented in the x, y, z

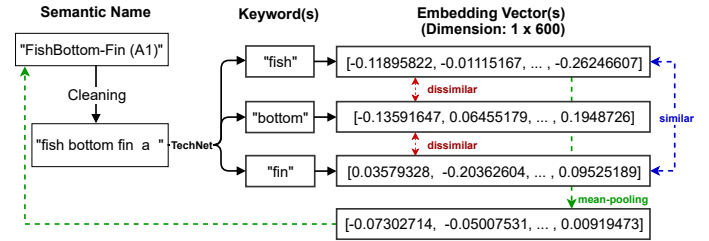


FIGURE 3. AN EXAMPLE OF FEATURE ENCODING FOR THE SEMANTIC NAMES OF AN ASSEMBLY.

coordinates). All physical properties are generated by Autodesk Fusion 360 during the user’s design process. The physical properties are normalized to the standard scale.

Geometric Information The 3D geometry of every body is used to render 2D depth images from 12 different views equally spaced around the object. A Multi-View Convolutional Neural Network (MVCNN) [46] is trained by feeding it the 2D images and enforcing it to classify the geometric shapes into 21 classes selected for the experiments. Once trained, the model’s weights are frozen, and the classification layer is truncated. The model is then used as a pre-trained encoder to compute 512-dimensional visual embeddings of all the available geometric data. The visual embeddings are used to represent the `body geometry`. The MVCNN is trained using the same train-test split as the GNN.

Connection Types Each assembly in the dataset comprises multiple interconnected bodies via various assembly relationships. Specifically, we consider three predominant types of connections. `Contacts` define the relationship between two bodies whose faces are in contact with each other. `Joints` define the relationship between two bodies whose relative pose and degrees of freedom are constrained. `Hierarchical edges` represent the user-assigned relative hierarchy of the bodies, indicating whether multiple bodies share the same occurrence. The connection types are encoded using the one-hot method as a categorical feature.

Global Features Each assembly has features that are shared globally among the bodies of the same assembly. These features include the assembly physical properties, assembly geometric properties, design category, industry, and products used to create the design. For more details see Appendix A.1.

3.3 Graph Construction

Each assembly is represented as a directed and attributed assembly multi-graph, where bodies that correspond to individual parts are represented as nodes. Connections that represent

assembly relationships are represented as edges. The graph is directed since each edge contains a source node, a destination node, and a direction. The graph is attributed since each node and edge contains encoded features. Furthermore, the graph allows multiple edges between any pair of nodes.

Graph Nodes Each body is represented as a node containing semantic name embeddings, visual embeddings, and physical properties. The encoded global features of the graph are concatenated with the node feature and shared across the nodes of the same graph. The ground-truth material names encoded as one-hot vectors are also considered graph node attributes and used in the partial algorithm-guided experiment.

Graph Edges The body nodes are connected through edges that represent assembly relationships between the bodies (Section 3.2). Specifically, `Assembly Edge` contains the one-hot encoding of connection type. Note that there can be multiple edges between a pair of nodes, corresponding to the multiple contact points between two bodies’ surfaces. If multiple bodies share the same occurrence in the assembly hierarchy, they are given a pair-wise `Hierarchical edge`.

Once the graphs are constructed, graphs with less than three nodes or less than two edges are discarded, resulting in 4210 valid assembly graphs with 85,089 nodes and 118,668 edges. The encoded semantic embedding, the physical properties, and the geometric information are concatenated into a node feature matrix for each graph. Similarly, the one-hot encoded connection types are concatenated into an edge feature matrix. The node and edge feature matrices and the connectivity information represented in coordinate (COO) format are organized into PyTorch Geometric [47] graph objects and are later fed into the GNN model.

3.4 Learning Framework

Given a set of graphs $\mathcal{G} = \{G_k\}_{k=1}^N$ where each graph $G_k = (V, E, \mathbf{X})$ consists of $|V|$ nodes, $|E|$ edges ($E \subseteq V \times V$), initial node features $\mathbf{X} \in \mathbb{R}^{|V| \times d_x}$ and edge attributes e_{uv} for $(u, v) \in E$, and the corresponding node labels $\{y_1^1, \dots, y_{|V|}^1, \dots, y_1^N, \dots, y_{|V|}^N\}$, the task of supervised node classification is to learn a representation $h_v, \forall v \in V$ such that the node labels can be predicted using the representations. GNNs use a neighborhood aggregation approach, where representation of node v is iteratively updated by aggregating representations of neighboring nodes and edges. After k iterations of aggregation, the representation captures the information within its k -hop neighborhood [22, 48]. Formally, the k -th layer of a GNN is defined as:

$$h_v^{(k)} = f_\theta^{(k)} \left(h_v^{(k-1)}, g_\phi^{(k)} \left(\left\{ h_u^{(k-1)}, e_{uv} : u \in N(v) \right\} \right) \right) \quad (1)$$

where $h_v^{(k)}$ is the representation of node v at the k -th layer and $N(v)$ denote neighbors of v . $f_\theta(\cdot)$ and $g_\phi(\cdot)$ denote parametric combination and aggregation functions. The choice of these functions can dramatically affect the expressiveness power of a GNN and different instantiations of them produce variants of GNNs such as GraphSAGE [49], Graph Convolution Network (GCN) [23], Graph Attention Network (GAT) [24], and Graph Isomorphism Network (GIN) [25].

The learning framework consists of a GNN encoder followed by a Multi-Layer Perceptron (MLP) network acting as a classifier head (Figure 1). The GNN encoder consists of a message-passing block in each layer followed by a Leaky Rectified Linear Unit [50] as the non-linearity. The final node embeddings are computed similar to Jumping Knowledge Networks [51] in which node embeddings computed in all the GNN layers are aggregated by summation. The final node embeddings are then passed through the MLP with batch normalization layers [52], Parametric Rectified Linear Unit (PReLU [53]) for generalized and learnable non-linearity activation, and a softmax activation for predicting class distributions. The weighted cross-entropy loss is adopted as the training objective to prevent the neural network from overlooking rare classes. The weights for each class are initialized as inversely proportional to the ground-truth class frequencies. The type of GNN layers is considered as a hyperparameter (see 4.1).

4 EXPERIMENTS

We evaluate and analyze the effectiveness of our framework with the Fusion 360 Gallery dataset. We designed three experiments through a variation of our base framework to address engineers’ diverse expertise, needs, and goals. We first introduce a general setup, including the results of hyperparameter tuning. Then we discuss the feature importance in material prediction through ablation experiments. Finally, we provide the detailed motivation, description, results, and discussion for each experiment. We then address the following questions: (1) Which features have the largest influence on material prediction? (2) How well can our method predict the ground-truth material when observing the top-1, top-2, and top-3 predictions? (3) How does our method perform when the material of some of the bodies in the assembly is known? (4) Can the user restrict the search space by guiding the network towards a specific material category for each body?

4.1 General Setup

In preparation for the training process, we split the constructed assembly graphs into a train set for model training, a validation set for parameter tuning, and a test set for performance evaluation based on the proportions of 56%, 24%, 20% respectively through a random sampling process. The assembly graphs

allocated to the test set are pre-defined by the Fusion 360 Gallery dataset and remain fixed across the dataset-splitting process of different experiments. We run each experiment for 10 iterations to account for variations and evaluate the performance based on the mean and standard deviation of the results. To evaluate the multi-class classification task of material prediction, considering the highly-skewed distribution of material labels (as shown in Appendix A.2), we report the *micro* F_1 score calculated by weighting each prediction instance equally.

To find the best set of hyper-parameters for the GNN, we ran a grid search on the number of GNN layers $\{1, 2, 3, 4, 5, 6, 7, 8\}$, the size of hidden dimensions $\{64, 128, 256, 512\}$, and the type of GNN layer $\{SAGE, GAT, GIN, GCN\}$ (as introduced in Section 2.2). After running each experiment configured with a combination of hyper-parameters for 10 iterations, we observed the best average performance and stability using 7 layers of GraphSAGE GNN with 256 hidden dimensions, where the formulation and advantage for the GraphSAGE GNN are described in Appendix B.3. Therefore, we adopted this configuration for the following experiments unless stated otherwise.

4.2 Feature Importance

Description To evaluate the importance of the features introduced to the graph construction and learning process, we perform an ablation study by systematically analyzing the variation of prediction performance by dropping sets of node and edge features. Specifically, we investigate the importance of node features by excluding only one node feature at a time during graph generation and learning. Subsequently, we re-run the node feature ablation experiments under the scenarios of no edge ablation and Hierarchical edge ablation to investigate the impact of edge features on the overall performance. Finally, we provide a set of baselines by including all node features.

TABLE 1. FEATURE ABLATION RESULTS, MICRO- F_1 SCORE

NODE ABLATION	EDGE ABLATION	
	NONE	HIERARCHICAL
BODY NAME	0.384 ± 0.02	0.416 ± 0.03
OCCURRENCE NAME	0.399 ± 0.01	0.423 ± 0.01
SEMANTIC NAMES	0.317 ± 0.05	0.373 ± 0.07
BODY PHYSICAL PROPERTIES	0.413 ± 0.02	0.425 ± 0.01
OCCURRENCE PHYSICAL PROPERTIES	0.415 ± 0.02	0.392 ± 0.06
BODY GEOMETRY	0.394 ± 0.02	0.420 ± 0.00
GLOBAL FEATURES	0.393 ± 0.04	0.429 ± 0.01
NONE	0.404 ± 0.02	0.425 ± 0.01

Results and Discussion Shown in Table 1 is the feature ablation results, where SEMANTIC NAMES indicates the body and occurrence names of the node, NONE and ALL indicate no feature and all features being ablated, respectively.

Focusing on the scenario in which *no ablation of edges is performed*, we observe that the most important node feature is the semantic names of the body since the exclusion of it significantly reduced the averaged performance (from 0.404 to 0.317) and increased the standard deviation (from 0.02 to 0.05). Dissecting the semantic names feature, we observe that both body name and occurrence name impact the accuracy when ablated, where the body name’s importance surpasses that of the occurrence name. Furthermore, excluding physical properties reflects an increase in averaged performance, which indicates that they are hindering the learning process. This phenomenon is unexpected since the physical properties of an assembly body should be correlated with its material choice. Therefore, we choose to re-run the ablation experiments without considering the hierarchical edges to investigate this adversarial effect.

When *ablating hierarchical edges*, we observe that the semantic names node feature shows the most significance, which is in accordance with the previous result. Consequently, we suggest that the user-defined semantic names of bodies in CAD models are the single most useful feature for material prediction. Further research should be performed to investigate more effective ways to extract and encode this feature. In addition, the physical properties of the body and occurrence demonstrate significance, which is contrary to the result when no edge ablation is performed. One possible explanation would be that the user-defined hierarchical edges are connecting bodies that may not be correlated with each other in terms of structural and physical properties. Accordingly, the introduction of hierarchical edges may misguide the neural network to learn representations of physical properties based on incorrect structural context. Therefore, we drop the hierarchical edges during subsequent experiments to mitigate this effect and minimize the computation overhead.

4.3 Fully Algorithm-guided Prediction

Description A long-term goal of design automation is to fully automate the material selection process without user input. This experiment evaluates how our network performs in a fully algorithm-guided scenario, where the ground-truth labels of material features for all nodes are left out during training and are re-introduced only during the validation and testing phases. Therefore, the GNN learns the node representations of the body nodes without access to ground-truth materials, aggregates the single-node representations obtained through message passing directly into a joint set of representations, and performs material prediction on all graph nodes at the same time. During the evaluation,

we treat the unseen assembly graphs as newly produced user designs and allow the trained model to output the top-1, top-2, and top-3 material class labels for all bodies based on the predicted probability. We denote this approach as the **fully algorithm-guided prediction** since the entire learning process is done by the learning framework independently without any supporting input from the user. The predictions may provide a set of suggestions for the user to select for further analysis or use.

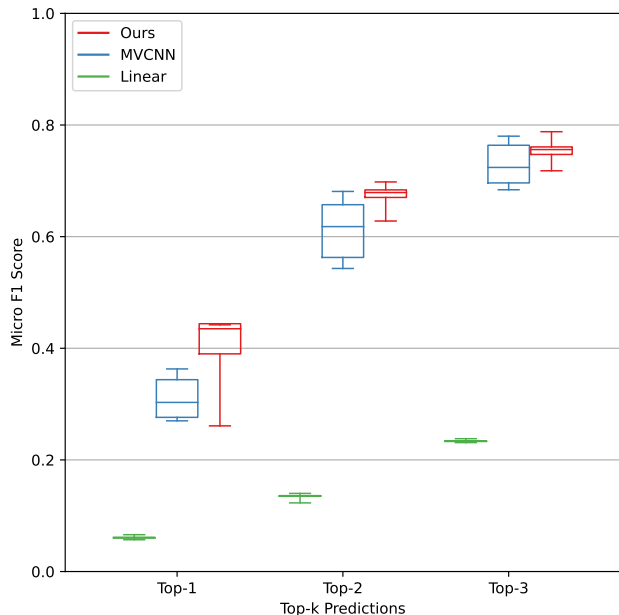


FIGURE 4. FULLY ALGORITHM-GUIDED PREDICTION RESULTS, COMPARED WITH SEVERAL BASELINES.

Results and Discussion As shown in Figure 4, we compare the material prediction performance of our proposed method with the baselines of MVCNN (which performs prediction based only on the geometry of bodies) and a linear regression model. When applying this work, the top-k method offers the flexibility to use a human-in-the-loop approach. In practice, this would streamline the decision process while leaving some creative decision-making to the experienced designer. Our method achieves a mean micro- F_1 score of 0.417 ± 0.04 , 0.677 ± 0.01 , 0.754 ± 0.01 for top-1, top-2, and top-3 predictions respectively, and outperforms the baselines in all top-k situations. This supports our hypothesis that material selection depends on factors other than the geometric features of the design. Designers choose the materials of one body in an assembly also based on the context of the rest of the assembly.

As more labels are considered in the prediction, the MVCNN approach appears to converge with our method, which is expected as two main classes dominate our dataset. Nonetheless, the application limits the MVCNN as it only learns from the geometry of prior designs, not from other multi-modal design data. In the remaining two experiments, we consider more input from the user, which can be easily introduced into the model.

4.4 Partial Algorithm-guided Prediction

Description Designers may have access to material labels for part of their assembly or design only one body in an existing assembly. To simulate this, this experiment introduces ground-truth material labels as node features into a portion of assembly graph nodes (*context* nodes), and the goal is to learn the labels for the rest (*target* nodes). During the message-passing process, the *context* nodes pass their embeddings through the edges to the *target* nodes, thereby updating their embeddings. This allows the GNN to perform material prediction on the *target* nodes by jointly learning from the topology of the graph and the ground-truth labels contained within the *context* nodes. During the evaluation, we limit the number of *context* nodes to a set of ratios and analyze the change in prediction performance jointly with a different number of GNN layers. We denote this as **partial algorithm-guided prediction** since the ground-truth labels provided by the user can influence, to a certain degree, the learning process and bias the final predictions. While the fully algorithm-guided prediction has the ground truth information of all nodes removed during training, the partial algorithm-guided prediction allows for a portion of context nodes to contain the ground truth labels during training but does not consider them during validation.

TABLE 2. PARTIAL-ALGORITHM PREDICTION RESULTS (MICRO- F_1 SCORE).

		Number of Layers							
		1	2	3	4	5	6	7	8
Context Nodes %	0.1	0.402	0.397	0.398	0.403	0.442	0.425	0.404	0.400
	0.2	0.386	0.421	0.414	0.435	0.416	0.450	0.389	0.385
	0.3	0.381	0.393	0.415	0.460	0.476	0.469	0.466	0.431
	0.4	0.402	0.399	0.435	0.462	0.475	0.477	0.474	0.459
	0.5	0.393	0.407	0.436	0.452	0.482	0.464	0.482	0.475

Results and Discussion Table 2 summarizes the results of the experiment, showing the averaged micro- F_1 score for the top-1 prediction. We observe that the performance increases as we

increase the ratio of augmented nodes and the number of GNN layers. The positive correlation between the ratio and the performance indicates that the model can constructively learn from augmented nodes’ topology and features during training and generalize at inference time. Similarly, the positive correlation between the number of GNN layers and the performance implies that the model learns better representations by considering the messages passed from distant nodes and edges in each graph. We observe a slight decrease in performance when layers exceed 7 due to over-parametrization. In summary, the partial algorithm-guided prediction achieves the best performance in top-1 prediction of 0.482 ± 0.02 when considering 50% augmented nodes and 5 GNN layers, indicating the method’s effectiveness in making predictions based on partial inputs from the user.

4.5 User-guided Prediction

Description The designer might want to restrict the search space to a specific material category for each body, such as “metal”, “ceramic”, “aluminum alloy”, or “thermoplastics”. With a **user-guided prediction** approach, the user provides the network with ground-truth material categories for all bodies, thus guiding the network towards groups of materials based on their expertise, ideas, and design goals. The motivation of this experiment, together with the prior partial algorithm-guided experiment, is to avoid the limitation of innovation by allowing the user to input their design information into the learning framework, thereby leading the design process. Due to the lack of newly generated data by designers, we simulate the user inputs via masking and involving ground truth information in the graph generation process. The configuration of this experiment follows from the format of the fully algorithm-guided prediction. However, to aid the learning process, we introduce the hierarchical material category label of the ground-truth material as a feature of each node.

The material categories are borrowed from the material library in Autodesk Fusion 360 and consist of up to three tiers in the hierarchy, where a deeper depth indicates a more detailed classification. For example, a body made of mild steel has a Tier 1 material category “Metal”, Tier 2 category “Ferrous”, and Tier 3 category “Carbon Steel”. During the evaluation of this experiment, we limit the hierarchy of ground-truth material categories to a set of depths and analyze the change in performance based on the top-1, top-2, and top-3 prediction accuracy.

Results and Discussion Summarized in Table 3 are the user-guided prediction results. We observe the general trend of increasing performance with hierarchical material category input, which validates our hypothesis that the user-input material class information can effectively guide the model in making predictions. One phenomenon to notice is the convergence of performance when increasing the hierarchy of material class informa-

TABLE 3. USER-GUIDED PREDICTION RESULTS.

TOP-K	MATERIAL CLASS			F ₁ SCORE	
	TIER 1	TIER 2	TIER 3	MICRO-(F_m)	WEIGHTED (F_w)
1	✗	✗	✗	0.417 ± 0.04	0.392 ± 0.03
	✓	✗	✗	0.546 ± 0.01	0.527 ± 0.01
	✓	✓	✗	0.736 ± 0.03	0.746 ± 0.03
	✓	✓	✓	0.731 ± 0.04	0.757 ± 0.03
2	✗	✗	✗	0.677 ± 0.01	0.630 ± 0.01
	✓	✗	✗	0.684 ± 0.10	0.677 ± 0.07
	✓	✓	✗	0.841 ± 0.11	0.851 ± 0.09
	✓	✓	✓	0.897 ± 0.01	0.903 ± 0.01
3	✗	✗	✗	0.754 ± 0.01	0.704 ± 0.01
	✓	✗	✗	0.781 ± 0.12	0.763 ± 0.10
	✓	✓	✗	0.889 ± 0.12	0.891 ± 0.12
	✓	✓	✓	0.953 ± 0.01	0.954 ± 0.01

tion with the number of prediction selections fixed. This indicates that the model can efficiently adapt to the user-input information, even if the information provided is minimal. Therefore, in practice, our framework can provide users with an accurate and compendious set of material selections guided by a limited amount of user-input material class information.

5 DISCUSSION

5.1 Limitations and Improvement

Class Imbalance As shown in Table 5 of Appendix A.2, the distribution of the ground-truth materials is highly skewed. Specifically, the default material (Steel) significantly dominates over the less common ones (i.e., Brass). The lack of representative examples from the minority class and the overwhelming amount of default cases less relevant to the task will misguide the model to make predictions biased towards the majority class. Despite the efforts made to mitigate the impact of imbalanced material labels, such as down-sampling the data, grouping rare labels, and adopting weighted loss calculation, the issue persists.

We seek to collect additional data with a more varied distribution of features and ground-truth materials for improvement. This could be achieved either by expanding the currently adopted Fusion 360 Gallery Assembly Dataset via inviting more designers to contribute or by using publicly available datasets such as ABC [9]. Furthermore, data augmentation through manipulating existing graphs, such as creating sub-graphs or synthesized graphs from past designs, may be a viable direction for improving the distribution of material classes. Existing tools designed

to support automated reasoning, such as Open-NARS [54], could also be applied to the graph representation learning process to support the inference and imputation of rare material classes. Furthermore, we are working on re-categorizing the hierarchical categories of material labels to produce a more reasonable distribution while preserving the soundness of the designs.

Functional and Behavioral Features Our current work mainly focuses on the *structural* aspect of material selection, such as the geometric and physical properties of bodies, as well as the assembly relationships that correlate them. While structural analysis has proven to be successful in the derivation and tabulation of material performance indices for standard mechanical design cases [55], the *functional* and *behavioral* aspects of the design are also highly influential to material selection [56, 1]. Therefore, one promising improvement would be to incorporate the functional information (i.e., the purpose) and the behavioral information (i.e., the attributes) of bodies as node features to narrow down the search space. While the TechNet embeddings of the semantic names might be implicitly representing some functional and behavioral aspects, introducing new features allows for a refined feature encoding process that can more effectively reflect the multifaceted information relevant to material selection in design automation, thereby enhancing the overall performance and robustness of the proposed framework when dealing with multi-modal real-world inputs.

5.2 Future Plans

Regression for Material Properties One of the limitations of the proposed framework is the dependency on a specific material library found in Autodesk Fusion 360. This dependency comes from framing the problem as a node classification task, where the number of possible classes is equal to the number of materials in the library, which for this dataset is 576. The categorical classification is challenging due to the number of possible classes. It also limits the application of the trained model to only materials found in the original dataset, which is incomplete and might not suit the needs of designers in different industries. A possible improvement might be to frame the problem as a regression task and develop a model to predict relevant physical and mechanical properties involved in the selection of materials, building on prior work that leveraged neural networks to predict the density and Young’s modulus of materials given their chemical compositions [57]. By mapping relevant material properties of the material library of the training data onto an Ashby chart, clustering of the material properties on the chart would enable a more flexible material selection method.

Graph Predictions Given the proposed framework’s ability to capture structural characteristics of CAD models via graph construction and learning relational information of design features

through graph representation learning, we envision the potential expansion of our work in producing diverse predictions for design automation that is not limited to material selection. Specifically, we plan to incorporate graph node prediction, graph edge prediction, and global context prediction experiments through a variation of our current graph representation learning process. For **graph node prediction task**, we aim to automate the user-design process of assembly bodies by providing suggestions regarding the structural, functional, and behavioral features through a joint representation learning of node, edge, and global graph features. The current material prediction task falls into node prediction and may serve as a relevant foundation. For **graph edge prediction task**, we aim to support the user-design process of organizing and correlating assembly bodies by providing insights into their hierarchical and relational information through representation learning based on their properties and that of the assembly. For **global context prediction**, we aim to provide users with an informative overview of the entire assembly design through representation learning tailored to the global graph features shared across individual bodies. The overview of the assembly design we provide may adapt to the introduction of newly designed bodies, thereby serving as a flexible auxiliary unit to address the needs of different users.

6 CONCLUSION

In this paper, we propose a unified framework that contributes to design automation by predicting the material of bodies in assemblies through graph representation learning, given user design knowledge input. Furthermore, we develop a systematic workflow for the feature extraction, encoding, and assembly graph construction of CAD models from the Fusion 360 Gallery Assembly Dataset.

During the experimental evaluation, we present three experiments through a variation of our base framework tailored to the diverse needs of designers. For the fully algorithm-guided experiment, in which predictions of all body materials are performed simultaneously without any ground-truth input, our model achieved a micro- F_1 score of 0.417, 0.677, 0.754 for top-1, top-2, and top-3 predictions, respectively, surpassing those of the MVCNN baseline. For the partial algorithm-guided experiment, in which predictions of target nodes are based on a joint representation learning of graph topology and the ground truths of neighbor nodes, our model achieved 0.482 top-1 micro- F_1 . For the user-guided experiment, in which the user’s input of ground-truth material categories take part in the learning process, our model achieved an averaged micro- F_1 score of 0.731 for top-1 prediction and up to 0.953 for top-3 predictions.

While our results demonstrate the feasibility of leveraging graph representation learning for feature predictions on graphically represented CAD models, the configuration of the three experiments shows promise in supporting human-in-the-

loop design automation applications. Specifically, we believe the proposed framework’s capacity to accommodate large-scale databases and flexibility in incorporating the designer’s knowledge can be used to create a recommendation system for users by learning best practices from existing designs. Furthermore, the framework may serve as a baseline for future works leveraging graph neural networks for design automation.

We view material selection as one step towards a fully automated data-driven design tool capable of synthesizing assemblies from design requirements. We hope to support further research to bridge the gap between the understanding of human designers and that of intelligent design agents.

ACKNOWLEDGMENT

This research was supported by the U.S. NASA MUREP Institutional Research Opportunity (MIRO) program (Award Number 80NSSC19M0200), donation from the Autodesk Inc, and partially carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration (80NM0018D0004).

REFERENCES

- [1] Arnold, S. M., Cebon, D., and Ashby, M., 2012. “Materials selection for aerospace systems”.
- [2] Mouritz, A., 2012. *Introduction to Aerospace Materials*. Woodhead Publishing in Materials. Elsevier Science.
- [3] Miller, B. A., 2002. “Materials Selection for Failure Prevention”. In *Failure Analysis and Prevention*. ASM International, 01.
- [4] Sarica, S., Luo, J., and Wood, K. L., 2020. “TechNet: Technology semantic network based on patent data”. *Expert Systems with Applications*, **142**.
- [5] Ferrero, V., DuPont, B., Hassani, K., and Grandi, D., 2022. “Classifying component function in product assemblies with graph neural networks”. *Journal of Mechanical Design*, **144**(2).
- [6] Siddharth, L., Blessing, L., Wood, K. L., and Luo, J., 2022. “Engineering knowledge graph from patent database”. *Journal of Computing and Information Science in Engineering*, **22**(2).
- [7] Sargent, P., 2016. *Materials information for CAD/CAM*. Elsevier.
- [8] Willis, K. D. D., Pu, Y., Luo, J., Chu, H., Du, T., Lambourne, J. G., Solar-Lezama, A., and Matusik, W., 2021. “Fusion 360 gallery: A dataset and environment for programmatic cad construction from human design sequences”. *ACM Transactions on Graphics (TOG)*, **40**(4).
- [9] Koch, S., Matveev, A., Jiang, Z., Williams, F., Artemov, A., Burnaev, E., Alexa, M., Zorin, D., and Panozzo, D., 2019. “Abc: A big cad model dataset for geometric deep learning”. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [10] Kim, S., Chi, H.-g., Hu, X., Huang, Q., and Ramani, K., 2020. “A large-scale annotated mechanical components benchmark for classification and retrieval tasks with deep neural networks”. In *Proceedings of 16th European Conference on Computer Vision (ECCV)*.
- [11] Willis, K. D. D., Jayaraman, P. K., Chu, H., Tian, Y., Li, Y., Grandi, D., Sanghi, A., Tran, L., Lambourne, J. G., Solar-Lezama, A., and Matusik, W., 2021. Joinable: Learning bottom-up assembly of parametric cad joints.
- [12] Jones, B., Hildreth, D., Chen, D., Baran, I., Kim, V., and Schulz, A., 2021. “Sb-gcn: Structured brep graph convolutional network for automatic mating of cad assemblies”. *arXiv preprint arXiv:2105.12238*.
- [13] Lambourne, J. G., Willis, K. D., Jayaraman, P. K., Sanghi, A., Meltzer, P., and Shayani, H., 2021. “Brepnet: A topological message passing system for solid models”. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12773–12782.
- [14] Jayaraman, P. K., Sanghi, A., Lambourne, J., Davies, T., Shayani, H., and Morris, N., 2020. “Uv-net: Learning from curve-networks and solids”. *arXiv preprint arXiv:2006.10211*.
- [15] Ashby, M. F., 1989. “Materials selection in conceptual design”. *Materials Science and Technology*, **5**(6), pp. 517–525.
- [16] Ullah, N., Riaz, A. A., and Shah, S. A., 2020. “Investigation on material selection for the columns of universal testing machine (utm) using granta’s design ces edupack”. *Technical Journal*, **25**(02), pp. 52–60.
- [17] Albiñana, J., and Vila, C., 2012. “A framework for concurrent material and process selection during conceptual product design stages”. *Materials & Design*, **41**, pp. 433–446.
- [18] van Kesteren, I., 2008. *Materials, user-product interaction and other decisions*.
- [19] Zarandi, M. H. F., Mansour, S., Hosseiniyou, S. A., and Avazbeigi, M., 2011. “A material selection methodology and expert system for sustainable product design”. *The International Journal of Advanced Manufacturing Technology*, **57**(9-12), pp. 885–903.
- [20] Abdullah, T. A., Popplewell, K., and Page, C. J., 2003. “A review of the support tools for the process of assembly method selection and assembly planning”. *International Journal of Production Research*, **41**(11), pp. 2391–2410.
- [21] Li, Y., Tarlow, D., Brockschmidt, M., and Zemel, R., 2015. “Gated graph sequence neural networks”. In *International Conference on Learning Representations*.
- [22] Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E., 2017. “Neural message passing for quantum chemistry”. In *International Conference on Machine Learning*, pp. 1263–1272.

- [23] Kipf, T. N., and Welling, M., 2017. “Semi-supervised classification with graph convolutional networks”. In International Conference on Learning Representations.
- [24] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y., 2018. “Graph attention networks”. In International Conference on Learning Representations.
- [25] Xu, K., Hu, W., Leskovec, J., and Jegelka, S., 2019. “How powerful are graph neural networks?”. In International Conference on Learning Representations.
- [26] Khasahmadi, A. H., Hassani, K., Moradi, P., Lee, L., and Morris, Q., 2020. “Memory-based graph networks”. In International Conference on Learning Representations.
- [27] Hassani, K., and Haley, M., 2019. “Unsupervised multi-task feature learning on point clouds”. In International Conference on Computer Vision, pp. 8160–8171.
- [28] Wang, T., Zhou, Y., Fidler, S., and Ba, J., 2019. “Neural graph evolution: Automatic robot design”. In International Conference on Learning Representations.
- [29] Sanchez-Gonzalez, A., Heess, N., Springenberg, J. T., Merel, J., Riedmiller, M., Hadsell, R., and Battaglia, P., 2018. “Graph networks as learnable physics engines for inference and control”. In International Conference on Machine Learning, pp. 4470–4479.
- [30] Sanchez-Gonzalez, A., Godwin, J., Pfaff, T., Ying, R., Leskovec, J., and Battaglia, P., 2020. “Learning to simulate complex physics with graph networks”. In International Conference on Machine Learning, PMLR, pp. 8459–8468.
- [31] Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E., 2017. Neural message passing for quantum chemistry.
- [32] Guo, K., and Buehler, M. J., 2020. “A semi-supervised approach to architected materials design using graph neural networks”. *Extreme Mechanics Letters*, **41**, p. 101029.
- [33] Marcheggiani, D., and Titov, I., 2017. Encoding sentences with graph convolutional networks for semantic role labeling.
- [34] Ahmed, F., Cui, Y., Fu, Y., and Chen, W., 2021. A graph neural network approach for product relationship prediction.
- [35] Tao, F., Sui, F., Liu, A., Qi, Q., Zhang, M., Song, B., Guo, Z., Lu, S. C.-Y., and Nee, A. Y. C., 2019. “Digital twin-driven product design framework”. *International Journal of Production Research*, **57**(12), pp. 3935–3953.
- [36] Feng, Y., Zhao, Y., Zheng, H., Li, Z., and Tan, J., 2020. “Data-driven product design toward intelligent manufacturing: A review”. *International Journal of Advanced Robotic Systems*, **17**(2), p. 1729881420911257.
- [37] Funkhouser, T., Kazhdan, M., Shilane, P., Min, P., Kiefer, W., Tal, A., Rusinkiewicz, S., and Dobkin, D., 2004. “Modeling by example”. *ACM transactions on graphics (TOG)*, **23**(3), pp. 652–663.
- [38] Williams, G., Meisel, N. A., Simpson, T. W., and McComb, C., 2019. “Design repository effectiveness for 3D convolutional neural networks: Application to additive manufacturing”. *Journal of Mechanical Design, Transactions of the ASME*, **141**(11), pp. 1–12.
- [39] Wang, Y., Grandi, D., Cui, D., Rao, V., and Goucher-Lambert, K., 2021. “Understanding professional designers’ knowledge organization behavior: A case study in product teardowns”. In International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Vol. 85420, American Society of Mechanical Engineers, p. V006T06A046.
- [40] Chaudhuri, S., Ritchie, D., Wu, J., Xu, K., and Zhang, H., 2020. “Learning generative models of 3d structures”. Vol. 39, Wiley Online Library, pp. 643–666.
- [41] Li, J., Niu, C., and Xu, K., 2020. “Learning part generation and assembly for structure-aware shape synthesis”. Vol. 34, pp. 11362–11369.
- [42] Mo, K., Guerrero, P., Yi, L., Su, H., Wonka, P., Mitra, N., and Guibas, L. J., 2019. “StructureNet: Hierarchical graph networks for 3d shape generation”. *ACM Trans. Gr.*, **38**(6).
- [43] Slyadnev, S., Malyshev, A., Voevodin, A., and Turlapov, V., 2020. “On the role of graph theory apparatus in a cad modeling kernel”.
- [44] Huang, H., Lo, S., Zhi, G., and Yuen, R., 2008. “Graph theory-based approach for automatic recognition of cad data”. *Engineering Applications of Artificial Intelligence*, **21**(7), pp. 1073–1079.
- [45] Sunil, V., Agarwal, R., and Pande, S., 2010. “An approach to recognize interacting features from b-rep cad models of prismatic machined parts using a hybrid (graph and rule based) technique”. *Computers in Industry*, **61**(7), pp. 686–701.
- [46] Su, H., Maji, S., Kalogerakis, E., and Learned-Miller, E. G., 2015. “Multi-view convolutional neural networks for 3d shape recognition”. In Proc. ICCV.
- [47] Fey, M., and Lenssen, J. E., 2019. “Fast graph representation learning with PyTorch Geometric”. In ICLR Workshop on Representation Learning on Graphs and Manifolds.
- [48] Hassani, K., and Khasahmadi, A. H., 2020. “Contrastive multi-view representation learning on graphs”. In International Conference on Machine Learning, pp. 4116–4126.
- [49] Hamilton, W., Ying, Z., and Leskovec, J., 2017. “Inductive representation learning on large graphs”. In Advances in Neural Information Processing Systems, pp. 1024–1034.
- [50] Maas, A. L., Hannun, A. Y., and Ng, A. Y., 2013. “Rectifier nonlinearities improve neural network acoustic models”. In in ICML Workshop on Deep Learning for Audio, Speech and Language Processing.
- [51] Xu, K., Li, C., Tian, Y., Sonobe, T., Kawarabayashi, K.-i., and Jegelka, S., 2018. “Representation learning on graphs with jumping knowledge networks”. In International Conference on Machine Learning, PMLR, pp. 5453–5462.

- [52] Ioffe, S., and Szegedy, C., 2015. “Batch normalization: Accelerating deep network training by reducing internal covariate shift”. In International conference on machine learning, PMLR, pp. 448–456.
- [53] He, K., Zhang, X., Ren, S., and Sun, J., 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification.
- [54] Wang, P., 2007. “From nars to a thinking machine”. In Proceedings of the 2007 Conference on Advances in Artificial General Intelligence: Concepts, Architectures and Algorithms: Proceedings of the AGI Workshop 2006, IOS Press, p. 75–93.
- [55] Ashby, M. F., 2011. *Materials selection in mechanical design*, 4th ed ed. Butterworth-Heinemann, Burlington, MA. OCLC: ocn639573741.
- [56] Cheng, Z., and Ma, Y., 2017. “Explicit function-based design modelling methodology with features”. *Journal of Engineering Design*, **28**, 02, pp. 1–27.
- [57] Merayo, D., Rodríguez-Prieto, A., and Camacho, A. M., 2020. “Prediction of physical and mechanical properties for metallic materials selection using big data and artificial neural networks”. *IEEE Access*, **8**, pp. 13444–13456.
- [58] Kingma, D. P., and Ba, J., 2017. Adam: A method for stochastic optimization.
- [59] Loshchilov, I., and Hutter, F., 2017. Sgdr: Stochastic gradient descent with warm restarts.
- [60] Hochreiter, S., and Schmidhuber, J., 1997. “Long short-term memory”. *Neural computation*, **9**(8), pp. 1735–1780.

A Appendix A: Dataset and Feature Distributions

A.1 List of Features Extracted from the Dataset

TABLE 4. LIST OF EXTRACTED FEATURES AND ENCODING METHODS.

Category	Design Feature	Description	Encoding Method	Feature Type
Semantic	Body name	Semantic name of the body.	TechNet	Node
	Occurrence name	Semantic name of the occurrence of the body.	TechNet	Node
Geometric	Body geometry	Geometry of the body in OBJ format.	MVCNN	Node
	Body physical properties	Properties of the body (center of mass, surface area, volume).	Float	Node
	Occurrence physical properties	Properties of the occurrence (surface area, volume).	Float	Node
Material	Material	The material of the body, treated as ground-truth label.	One-hot	Node
	Material class	The hierarchical classification of the material, treated as a user-introduced feature only in the user-guided experiment. From <i>tier 1</i> (broad) to <i>tier 3</i> (specific).	One-hot	Node
Global	Assembly physical properties	The physical properties of the overall assembly (center of mass, volume).	Float	Global
	Assembly geometric properties	The geometric properties of the overall assembly (total number of edges, faces, loops, shells, vertices).	Integer	Global
	Design category	The category specified by the user (automotive, art, electronics, engineering, game, machine design, interior design, product design, robotics, toys, etc.).	One-hot	Global
	Industry	The industry specified by the user to describe the assembly (architecture, engineering & construction; civil infrastructure; media & entertainment; product design & manufacturing; other industries).	One-hot	Global
	Products	The products used to create the design.	One-hot	Global
Hierarchical	Hierarchical	The designer-defined hierarchy, between bodies that share an occurrence.	One-hot	Edge
	Joints	Constraints defining the relative pose and degrees of freedom between a pair of occurrences.	One-hot	Edge
	Contacts	Faces that are in contact between different bodies.	One-hot	Edge

A.2 Statistics of Material Categories

TABLE 5. LIST OF THE TOP-20 MATERIAL CATEGORIES, THEIR COMMON NAMES, AND COUNTS.

Material id	Material name	Count
PrismMaterial-018	Steel	40054
PrismMaterial-022	ABS Plastic	2657
PrismMaterial-002	Aluminum	2622
Prism-112	Plastic - Glossy (Black)	2435
Prism-052	Gold - Polished	2286
MaterialInv_029	Steel, Mild	1985
Prism-027	Aluminum - Polished	1954
Prism-089	Paint - Enamel Glossy (Black)	1593
Prism-094	Paint - Enamel Glossy (White)	1361
Prism-047	Chrome	1262
Prism-029	Aluminum - Anodized Glossy (Blue)	1235
Prism-113	Plastic - Matte (Black)	1226
Prism-322	Bamboo Light - Semigloss	1206
PrismMaterial-017	Stainless Steel	1110
Prism-042	Brass - Polished	1050
Prism-126	Plastic - Glossy (White)	1042
Prism-256	Steel - Satin	1019
PrismMaterial-003	Brass	997
Prism-127	Plastic - Matte (White)	987
Prism-230	Iron - Polished	967
-	Other materials	62254
Total		131302

A.3 Statistics of Constructed Graphs

The statistics of the constructed graphs are as follows. Each graph has an average of 24 nodes, a maximum of 821, and a standard deviation of 43.1. Each graph has an average of 40 edges, a maximum of 2336, and a standard deviation of 79.9.

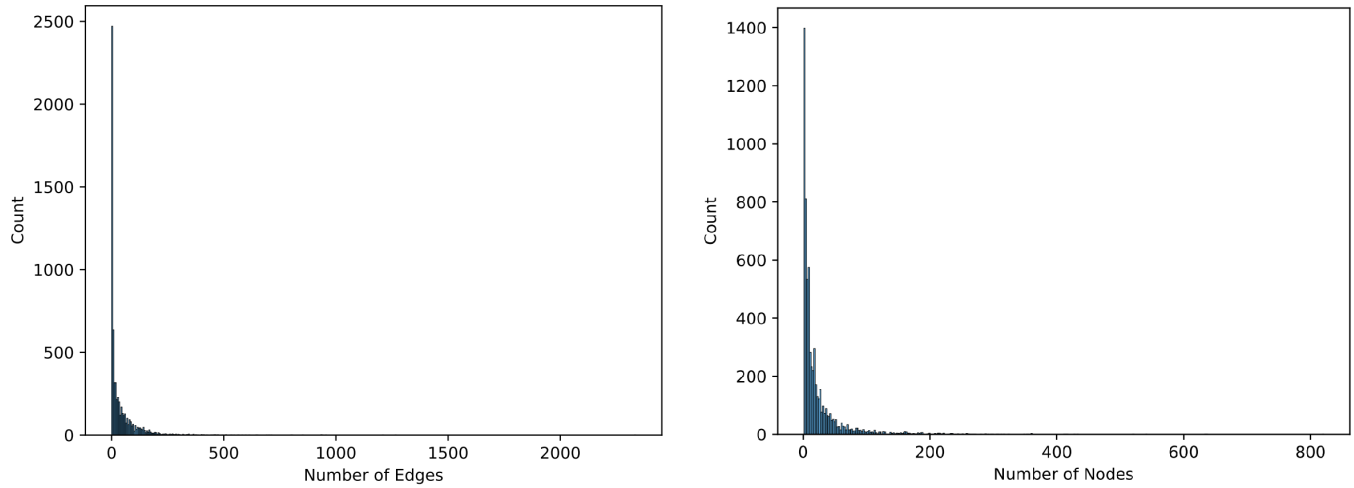


FIGURE 5. Distribution of Graph Nodes and Edges.

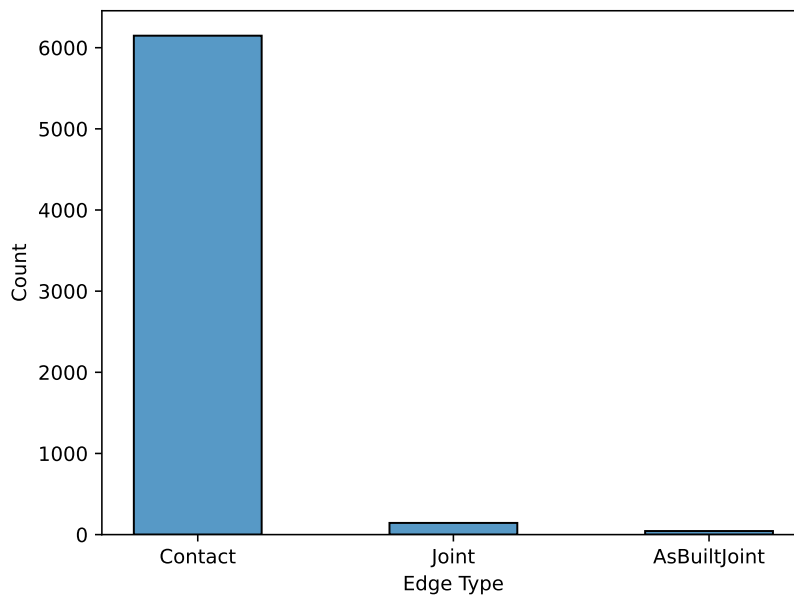


FIGURE 6. Distribution of Graph Edge Types.

B Appendix B: Additional Explanation of Methods

B.1 Feature Extraction

As introduced in Section 3.1, most features of the assemblies are encapsulated inside their corresponding JSON files and ordered in the format of nested dictionaries. Therefore, we devised the following process to extract the features of each body and their corresponding correlations in the assembly. These extracted features were further encoded following the methodology in Section 3.2 and used for graph construction in Section 3.3.

Hierarchical bodies For each occurrence in the dictionary’s `tree`, we search the bodies within and extract their body features by tracking their UUIDs. After filtering out the bodies that are indicated as invisible, we combine each remaining body feature and their corresponding occurrence features as its final features. Note that the occurrences in “tree” may have a nested structure (i.e., each occurrence may have sub-occurrences) and are therefore hierarchical. To reflect this, we recurrently perform the searching process and assign each body with a hierarchical depth parameter. These features are used later for node creation.

Assembly relationship For each connection type (`joints`, `as-built joints`, `contacts`) in the JSON file, we search the entities within that contain a body and a corresponding occurrence that are visible. We match these bodies via their UUID to the root and hierarchical bodies we extracted prior to this step and pair them up with their connection type. These features are used later for edge creation.

B.2 GNN and MVCNN Training Configurations

For the GNN training process, we adopted the Adam optimizer [58] with the learning rate initialized to be 0.001 and a cosine annealing scheduler [59] with the maximum iteration set to the number of epochs to tune the learning rate for minimizing the loss. After obtaining the embedding from each GNN layer, we apply the Leaky Rectified Linear Unit with a negative slope of 0.2 for non-linear activation.

The MVCNN is trained using a PyTorch implementation and uses the ResNet architecture with a supervised learning regime. A patience factor is used to stop the training process after 20 epochs that see an increase in the validation accuracy, resulting in around 30 training epochs. The models are trained with a batch size of 8, 512 embedding dimensions, 12 views, and $1e-4$ learning rate.

B.3 GNN Layer Formulations

As discussed in Section 3.4 and Section 4.1, we experimented with the GNN layer from the set of $\{SAGE, GAT, GIN, GCN\}$ through a variation of the combination and aggregation functions. After tuning the hyperparameters, we adopted the GraphSAGE neural network, which is particularly suitable for the embedding generation of un-seen nodes on large and diversely structured graphs through sampling and aggregating neighborhood information. The formulation of the parametric combination function $f_{\theta}(\cdot)$ and the aggregation function $g_{\phi}(\cdot)$ for the GraphSAGE network at layer k are as follows.

GraphSAGE [49] The aggregation and combination processes for GraphSAGE are formulated in Equation 2. For the first equation that formulates the aggregation process, $a_v^{(k)}$ indicates the aggregated feature for node v at layer k , $LSTM$ indicates a Long Short-Term Memory aggregator [60], h_u^{k-1} indicates the node feature for node u at layer $(k-1)$, and $N(v)$ indicates the set of nodes neighboring to node v . For the second equation that formulates the combination process, $ReLU$ indicates a Rectified Linear Unit, \parallel indicates matrix concatenation operation, and W^k represents a weight matrix for layer k that is learnable. GraphSAGE allows for the efficient generation of embeddings for un-seen nodes through sampling and aggregating neighborhood information and is suitable for representation learning on large graphs.

$$a_v^{(k)} = LSTM(\{h_u^{(k-1)}, u \in N(v)\}), \quad h_v^{(k)} = ReLU(W^{(k)}[h_v^{(k-1)} \parallel a_v^{(k)}]) \quad (2)$$