# Principles and Metrics for Curating Large Engineering Simulation Data Sets for ML

**Jami J. Shah, Satchit Ramnath, Stefan Menzel, Thiago de Jesus de Araujo Rios, Fatma Kocer, Eamon Whalen, Joseph Pajot, Alex Adrian, Prakash Kumar**

**2025**

**Preprint:**

This is an accepted article published in ASME Journal of Computing and Information Science in Engineering (JCISE). The final authenticated version is available online at: https://doi.org/10.1115/1.4070033

# Principles and Metrics for Curating Large Engineering Simulation Data Sets for ML

Jami J. Shah, The Ohio State University,
Satchit Ramnath, Clemson University
Stefan Menzel, Thiago Rios, Honda Research Institute Europe
Fatma Kocer, Eamon Whalen, Joseph Pajot, Altair Engineering
Alex Adrian, GE Aerospace
Prakash Kumar, Amazon Inc.

## Abstract

It is time to talk about data in its own right, not just its usage! Machine Learning applications are using a wide variety of data sources, some real, such as data collected by sensors and cameras in driving, and some artificial, such as data generated through numerical simulations. The latter mode has been gaining rapid popularity for engineering design and analysis. We are now seeing the beginnings of publicly shared data sets. Thus, the quality and efficacy of such data need to be considered before their use. In this paper, we attempt to outline systematic principles and quantifiable quality and efficacy metrics based on insights gained collectively from both data curation projects and usage of large engineering data sets. Specifically, this paper addresses issues related to generating BIG datasets from CAD and FEA: *Granularity* and *Modality*, applicable to both input and output data; *Variety* and *Balance*, applicable to input data; also, *Efficacy* for ML. Generation of Big Data by simulation requires the use of commercial CAD and FE packages, which poses multiple challenges: automation, integration and balancing sample variants. We propose Parametric Variety and Balance Matrices to study over or under representation of input attributes. Even if we have a large data set with good balance, it may not be suitable for ML if we do not see significant variation in response or performance variables. Several data generation, curation and utilization case studies are included in a variety of domains (aero, structural, thermal, manufacturing) and usage of metrics is demonstrated.

## 1. Introduction

The last decade has seen a rapid evolution of AI/Machine Learning applied to virtually every field of engineering. A primary driver appears to be the potential for democratization of complex engineering analyses (structural, thermal, aero, etc.) which could be placed in the hands of designers, rather than simulation experts, to get quick results for exploring designs. Training ML algorithms requires data sets which can potentially come from many sources, such as experiments, product usage, shop floor sensors or artificially generated by simulations. There is considerable up-front investment before benefits of ML can be realized. Engineering communities are engaged in tremendous efforts to generate data sets for a wide variety of applications. However, such efforts must not be carried out in an ad-hoc manner. It is necessary to create systematic methods and procedures. A fundamental understanding of the characteristics of datasets must be identified and quantified in order to determine the adequacy and suitability for targeted objectives. The design and evaluation of such datasets is the focus of this paper.

Data Curation for ML refers to the generation, structuring, labelling and integration of data collected from various sources. The scope of this paper, however, is limited to data collected from numerical simulation. As opposed to datasets of random objects [1]-[4], engineering data sets must consist of objects that are functional equivalents. For example, structures designed to carry the same loads, within the same

environmental conditions and spatial constraints. The designs generated in the GE Bracket design competition would meet this criterion [5].

Data curation cannot be done in a vacuum – it must be done in conjunction with project objectives and ML algorithm capabilities. Common engineering objectives are structural design/optimization, manufacturability evaluation, cost estimation, fluid and thermodynamics, etc. Unlike millions of unrelated random geometric objects stored in open source sites, such as *OnShape*, engineers are interested in analyzing or selecting from component geometries designed for the same functional requirements. Thus, specialized data sets must be curated to contain design variants that meet the same constraints and meant to operate under the same initial conditions or boundary conditions. The training data must have performance characteristics of each variant, obtained by simulations performed in a consistent manner. So we have two sides of the data: input (characteristics of the design variants) and response (performance attributes) in simulated datasets.

Conventional product design process typically iterates between design and analysis to find satisficing designs. No dataset is needed, although experience with past design can indirectly guide human designers and analysts. ML based product development requires a dataset in the specific domain and functional requirements, containing design variants and their performance predicted from simulations, and ML algorithm trained on that dataset. This allows data based prediction of performance of proposed candidate designs, as long as they are within the domain of the trained algorithm.

In engineering applications, ML is being used for classification, clustering and performance prediction. Generative AI is also an active area of research; ML can perform “interpolation” well but “extrapolating” is an open issue - depends on which data sources have been used to train the model and “how far off” one would like to go.

With the advances in artificial intelligence, machine learning (ML) has become a powerful technique to support the engineering process and search for innovative solutions. Depending on the application at hand, ML algorithms can be utilized as regression or classification models, e.g., to accelerate optimization processes through performance prediction [6][7], as well as to perform data analysis and to enhance the interpretation of solutions [8][9]. However, besides the nature of the task, the characteristics of the data, e.g., modality and size, constrain the selection and performance of ML algorithms. Hence, matching application and task characteristics is an important step prior to developing ML algorithms or engineering data sets.

Sarker[10] presents a comprehensive analysis of ML algorithms with respect to the task objective and type of utilized data. Although application-agnostic, the analysis in [10] can be extended to design and optimization tasks in engineering. A typical motivation for using ML in design optimization is to substitute computationally expensive simulation models by surrogate prediction (regression) methods based on the design parameters and historical data[7][11], which are often structured in tabular format with mixed integer- and real-valued variables. In another application scenario, particularly in robotics and human-machine cooperative tasks, researchers utilize reinforcement learning strategies, which iteratively apply the ML model in a simulation of the application scenario and adjust the model parameters based on the success rate of the model in the targeted task [12][13][14].

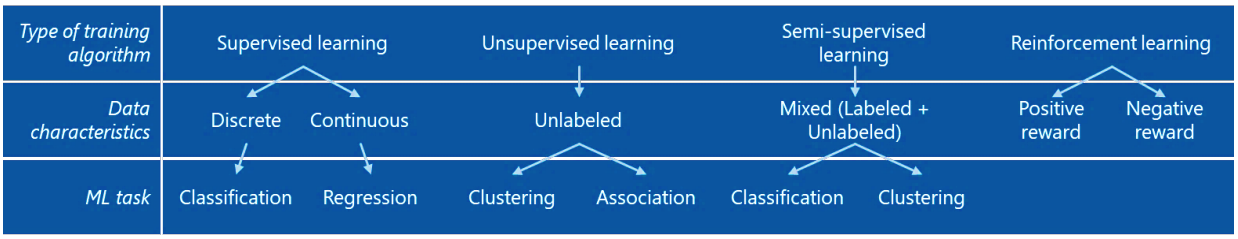



*Figure 1 – Taxonomy of types of ML algorithms based on data characteristics and task objectives as proposed in[10].*

With respect to design processing tasks such as shape modification and optimization, the introduction of geometric deep learning (GDL) algorithms enabled the development of shape-generative models using learning-based geometric representations. Among the GDL algorithms, (variational) autoencoders learn low-dimensional vector representations directly from geometric data, e.g., volumetric meshes and 3D point clouds, by compressing the data through a bottleneck layer, the so-called latent space. Differently from tabular data, geometric data are unstructured and often invariant to permutations, which hinders the development of ML architectures, but allows to learn more complex data structures. Hence, the design variables learned by GDL architectures neglect semantic information about the designs in the data set and allows one to seamlessly generate shapes with different topology, which is often a limitation for other representation methods [15]. Furthermore, one can also train ML models for performance prediction based on the latent variables in design optimization or exploration tasks [6], identify shape commonalities and relevant geometric features [16], among other data analysis tasks.

Simulation datasets are generated by creating variants of topology, geometry, parameters and other attributes, such as material properties. Each of the variants then must be put through the appropriate simulation process to determine performance factors of interest, such as stress, deflection, max. temperature, lift, drag, etc. What are the requirements for dataset to be appropriate for ML? This is a very difficult question from a general point of view: data set size, distribution of data, modality, optimal number and selection of features reflecting the use case. These are the questions explored in this paper.

## 2. Public Datasets in Engineering

Although advances in ML algorithms lead to better outcomes, the availability of high-quality datasets is an important part of this process. Generating high-quality training datasets, both labeled and unlabeled, for various ML algorithms can sometimes be difficult and expensive to produce [17]. Simulated datasets can be represented in various formats and modalities like text, image, sound, and 3D models (mesh, STL, BRep, etc.). A sampling of simulated engineering design datasets published for public use in the past few years is presented below. Obviously, this is by no means comprehensive, but meant to be representative.

Fusion 360 Gallery Dataset: The Fusion 360 Gallery Dataset contains rich 2D and 3D geometry data derived from parametric CAD models. The dataset is produced from designs submitted by users of the CAD package Autodesk Fusion 360 to the Autodesk Online Gallery. All CAD models are saved as **.f3d* format, which is Fusion 360's native file type. However, users can also export models to widely supported formats like *.STEP, *.IGES, *.STL, etc. The dataset provides valuable data for learning how people design, including sequential CAD design data, designs segmented by modeling operation, and assemblies containing hierarchy and joint connectivity information[19][20].

BIKED is an assortment of 4512 manually-designed bicycle models span not only common bicycle frames, but also unique models from remote corners of the bicycle design space. BIKED contains user-specified bicycle class data, which can be used for classification tasks and is great for visualizing the data. Shown below is a T-Distributed Stochastic Neighbor Embedding plot of the parametric bicycle design space. Corresponding images from models in different regions of the embedding are showcased[21].

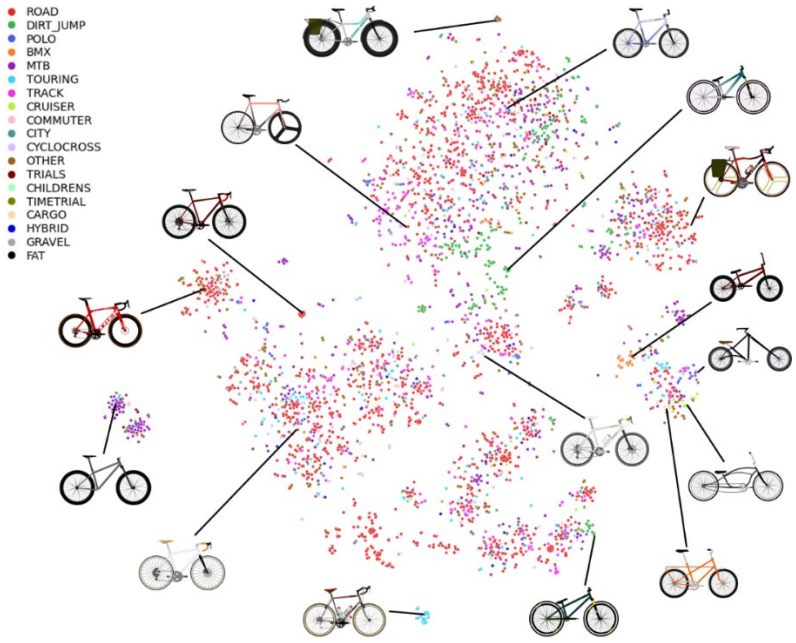



*Figure 2: T-Distributed Stochastic Neighbor Embedding plot[21]*

FRAMED explores data-driven methods for performance-aware design of bicycle frames. It contains structural performance quantities for 4500 bicycles, that include simulated results of each design under in-plane, transverse, and eccentric loading conditions. Results included are a variety of stresses, safety factors, and deflections in addition to model mass [22]. FRAMED and BIKED are not directly related.

2D Beam Topologies is a dataset of optimized topologies for a 2D beam. It is generated using an open-source solver for topology optimization based on the SIMP approach. It contains images of topology solutions of 10,000 randomly stated problems. The constraints and loads for each problem are sampled from a Poisson distribution. The figure below shows a subset of the designs in the dataset [23].

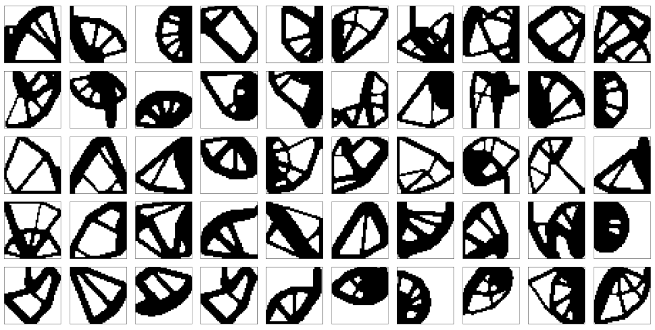


*Figure 3: Subset of TO designs in the dataset [26]*

TopologyGAN: A SIMP-based topology optimization framework was used to generate a data set. The 2D domain consists of a 64 × 128 grid structure consisting of square elements. A total of 49,078 optimized

structures were generated. This dataset was used to train a TopologyGAN model that utilizes the von Mises stress and strain energy density fields, and a modified residual network to predict the results for a given topology optimization problem[24].

CarHoods10k: The CarHoods10k data set comprises a set of over 10,000 3D mesh geometries for variants of car hood frames, generated through an automated, industry-grade Computer Aided Design (CAD) workflow. The data set provides realistic designs that were validated by experts with respect to realism, manufacturability, variability, and performance. Variations in geometries were generated by a feature-based approach that varies parameter values describing design features on parameterized base geometries (100 shape variants x 100 size variants/shape). Parameters describe feature patterns such as cut-outs or ribs on the hood frame and their properties. Geometries are represented as surface meshes (STL files) and are provided with the corresponding design parameter values and performance metrics generated using finite element analysis (FEA)[25].

SimJEB: The Simulated Jet Engine Bracket Dataset (SimJEB) is a collection of finite element simulations of crowdsourced engine bracket designs. This open dataset has several potential uses, including testing geometry processing and surrogate modeling algorithms and studying structural design strategies. The bracket designs originate from the "GE Jet Engine Bracket Challenge" and have been cleaned, oriented, scaled, meshed, and simulated. Each entry has a corresponding CAD file, tetrahedral mesh, triangular surface mesh, and structural simulation results from four load cases [5].

ShapeNET: a rich, annotated, large-scale dataset of 3D shapes provides data to enable research in computer graphics, computer vision, robotics, and other related disciplines. It consists of two subsets (i) *ShapeNetCore* and (ii) *ShapeNetSem. ShapeNetCore* is a subset of the full *ShapeNet* dataset with single clean 3D models and manually verified category and model position/orientation annotations. It covers 55 common object categories with about 51,300 unique 3D models. *ShapeNetSem* is a smaller, more densely annotated subset consisting of 12,000 models spread over a broader set of 270 categories [26].

Mechanical Components Benchmark: MCB is a large-scale dataset of annotated 3D objects of mechanical components that enables data-driven feature learning for mechanical components. It allows for exploring the shape descriptor for mechanical components which is essential to computer vision and manufacturing applications. The annotated dataset is represented in three formats, namely: (1) point clouds, (2) volumetric representation in voxel grids, and (3) view-based representation[27].

ModelNet: Provides researchers in computer vision, computer graphics, robotics and cognitive science, with a comprehensive clean collection of 3D CAD models of objects. To build the core of the dataset, a list was compiled of the most common object categories used by a human, and then collected 3D CAD models belonging to each object category. Human input was used to categorize each CAD model [28].

Aircraft Lift and Drag Dataset: Dataset of 3D CAD models of over 4045 aircraft designs, along with the performance metrics (lift-to-drag ratio) extracted from the ShapeNet Airplane Category. The dataset contains a collection of 3D aircraft models across various types, including commercial jets, military jets, private planes, and helicopters [52].

a. **Current state of datasets: Critique**

So many datasets have become publicly available in just a few years and more continue to come on line., that it is difficult to do justice. A comparison of the ones sampled above is shown in Table 1.

*Table 1: Attributes of selected geometry datasets for engineering applications*

| Dataset | Domain | Geometry rep | Data Size | Labelled? | Labels (if applicable) |
|---|---|---|---|---|---|
| FRAMED | structural | STL | 4500 | Labeled | Deflection, stress, mass |
| BIKED | structural | Images | 4512 | Labeled | Bike class labels |
| 2D Beam Top | structural | Images | 10,000 | No | - |
| TopologyGAN | structural | STL; Point Clouds | 49,078 | Labeled | Topological shape labels |
| CarHoods10k | structural | Catia & STL | 10,000 | Labeled | Deflection, stress, mass |
| SimJEB | structural | Mesh/STL | 381 | No | - |
| ShapeNETCore | General | Proprietary CAD | 51,300 | Labeled | Object class |
| ShapeNetSem | General | Proprietary CAD | 12,000 | Labeled | Object class |
| Mech Comps | Std. elements | Point clouds; STL | 58,696 | Labeled | Object class |
| ModelNET | General | Voxels | 15,000 | Labeled | Object class |
| Aircraft L/D Dataset | aerodynamics | ShapeNet CAD | 4045 | Labeled | Lift, Drag co-effs |

Each has defined its own criteria for design objective, data size and modality. Geometry is represented as images, mesh, STL, 3D CAD in various proprietary formats. Some are specific to particular applications (structural, aero), while others have no performance values/labels. Some that do have performance, like 2D beams, contain a mix of different load conditions not directly comparable. Even though the number of designs in these sets is impressive, it is not always clear how rich the set is in terms of feature balance and variety, key measures of data efficacy or even suitability for any type of ML algorithm. A practical issue with public datasets is that once published, they can be downloaded and used without any knowledge about who is using it, for what purpose and with what extent of success, unless that information is disclosed via publications. Towards that end we have included some ML case studies using engineering datasets in Sec 7.

## 3. Key Attributes, Characteristics of Simulation Datasets

### a. Data formats

Although ML datasets can come from many different sources, this paper is only concerned with artificially curated data from CAE simulation. Regardless of the physics domain, a pre-requisite for simulation is CAD geometry of components or assemblies. Components may be represented by Line, Surface, Solid Bodies or a combination of them. They may be native formats (*.CatPart, .agdb*, etc.), kernel formats (*.sat*, *.xt, .xb*), or standard data exchange format (STEP, IGES). The complexity of these data structures makes them difficult to process by ML. It is common to convert them to simpler representations, such as triangulated meshes (STL, OBJ), Point Clouds or even 2D/3D images, pixels or voxels.

CAE data may include model setup and process data (mesh, element type, BCs, solver settings) and results (deformed geometry, stress, strain, temperature, velocities, etc.). However, one can eliminate the need for process data if model settings are kept uniform for all parts.

If the dataset involves assemblies, or multi-step simulation (e.g., stamping + joining), representation becomes challenging, both of CAD and CAE data. The case study in Sec. 7d will overview such an application.

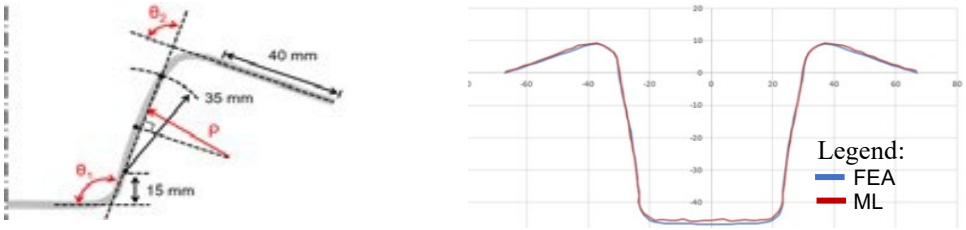
### b. Data modality

Data modality is the way or mode in which data exists. Geometry can be described textually with key attributes, feature lists or graphs, wireframe, key cross-sections or key parameters (thickness, aspect ratio,

curvatures). Parametric invariants can be expressed in terms of geometric dimensions and material properties. This can be listed in Tables, giving higher order description than “dumb” point clouds, leading to insights. The choice of modality is highly dependent on targeted application and what can be supported by chosen ML algorithms.

c. **Granularity of datasets**

Data granularity refers to the level or scale of detail present in a set of data, from abstract to fullest possible detail. In structural analysis one could simply archive key results, such as the maximum deflection or vM stress; in thermal, one could similarly keep only max temperature. This may be deemed as low granularity. On the other hand, the entire stress, strain, deflection, temperature field could be saved to the dataset, or high granularity. Below is an example of sheetmetal stamped hat section after springback at release from tooling. NUMISHEET benchmark [29] for this geometry has characterized the deformed geometry with just 3 key variables, two angles and a radius. ML could be trained to predict just these 3 variables. Alternatively, the entire deflected cross-section shape can be captured, node by node (high granularity), and used in training.




(a) *3 parameters characterization* *(b) full nodal deflection field*

*Figure 4 Stamped cross-section representation at varying granularity*

d. **Data Quality, Accuracy**

Performance responses are generated by repeated simulations on product variants. Attention must be paid to the *accuracy* and *uniformity* of simulations. Accuracy of simulations in light of geometry idealization requires comparing baseline designs, idealized vs original (detailed, un-idealized). The performance of baseline designs needs to be verified by experimental data or prior simulations which have been verified (ground truth). The usual good practices need to be followed to determine if reference simulations make sense, such as deflected shape, heat flow, etc.

Simulation *uniformity* or consistency refers to CAE setup, process, mesh density and boundary conditions (BCs). There are those who believe that mesh density should be kept constant in generating simulation datasets. This is probably not a good idea if component sizes vary a great deal, or if geometries have some very small and some very large features. Instead it is better practice to use mesh convergence as the determining factor for setting mesh size. Translating physical BCs to FE BCs can be challenging, in general. How can one apply BCs in a uniform way, even when geometry variants have different features at loads and supports? For example, in automotive hood support frames, hinge and lock designs can vary from hood to hood. These are also the locations for supports. How to apply BCs in a uniform way independent of the detailed hinge, lock features, if one is not interested in local behavior? Clearly, one needs to standardize supported DoFs, Load locations, area size and shape, as shown in Figure 5: Idealization/standardization of supports Figure 5.

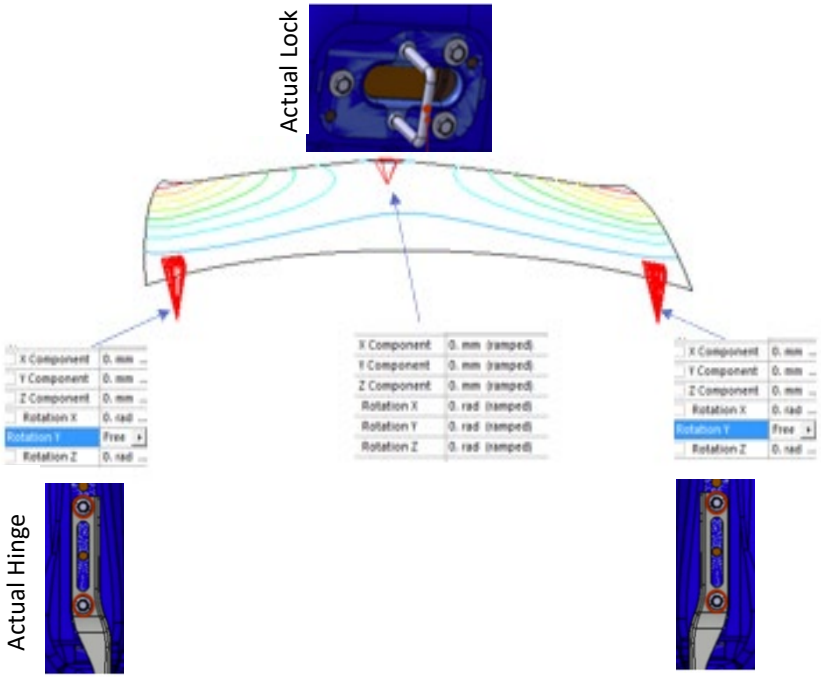



*Figure 5: Idealization/standardization of supports*

## 4. Efficacy of datasets for ML

To state the obvious, not every dataset will be effective for training every type of algorithm for desired objectives. Characteristics discussed in the previous section, modality and granularity, have to be suited to the objective and algorithm design. In addition, data size, parameters/features included and variety and balance of those features present in the dataset will no doubt influence training efficacy.

### a. Size of dataset

How much data is needed for ML? No one can answer this question in any definitive terms. It depends on the complexity of the problem, physics, number of input variables and responses. In one study, ALTAIR demonstrated that prediction accuracy can be increased progressively by using more data up to a point, beyond which there is marginal or no benefit [30]. So, at best, it is a case by case, trial and error process to arrive at a suitable training size, although it would no doubt be influenced by data variety and balance.

Since this paper is focusing on generating data by simulation, it is within the Curator's control to determine which features to include, parameters to vary, their respective ranges and levels (values). Therefore, it is possible to compute the size of the full factorial set as $\prod_{i=1}^{N_T} L_i$ for $N_T$ factors (input parameters/features), each of which have *L* levels (values/variants). Conducting these many simulations is usually cost and time prohibitive, so most opt for fractional factorial sets sampling. Suppose one decides on *S* simulations, then it is possible to get a sense of how well the design space is sampled with ratio *R*

$$R = S \Big/ \prod_{i=1}^{N_T} L_i$$

The above is only valid for components, not assemblies. When forming assemblies from random part simulations, geometrically incompatible parts need to be accounted for. In the denominator, physically incompatible combinations $I_c$ need to be subtracted, for a modified measure *R'*.

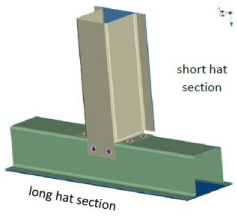
$$R' = S \Big/ \left(\prod_{i=1}^{N_T}(L_i) - I_c\right)$$

Since probability distributions are unknown, confidence limits cannot be determined from R or R' alone.

Usage of this metric is illustrated here by the example of stamped welded assemblies of two parts shown below. The Table below shows the five parameters and number of levels for component simulations. Based on full factorial combination, $\prod_{i=1}^{N_T} L_i$ would yield 1440 and 288 simulations. If the two components could be assembled randomly, we would get 1440*288 = 414,720 assembly combinations. However, compatibility of assembly and joining requires that the width of the bottom component match the depth of the top component. In this case, incompatible combinations $I_C$=(1440/20)*288= 393,984. If number simulated S =1000 (random), then assembly sampling ratio R' = 1000/(414720-393,984) = 4.8% .

Even though this metric does not tell us if this is a large enough data size for the intended purpose, it does give us a quantification of how much the design space was sampled.

*Table 2: Assembly combinations*




| Parameter | Width | Depth | Thickness | Material | BHF | $\prod L_i$ |
|---|---|---|---|---|---|---|
| Vertical Component | 4 | 20 | 3 | 2 | 3 | 1440 |
| Horz Component | 4 | 4 | 3 | 2 | 3 | 288 |

b.**Heterogeneity in data**

Dataset size alone is not sufficient to judge the efficacy; it is necessary to also look at balance and variety. Unbalanced or skewed datasets will be poor candidates for ML training. Two heterogeneity criteria are introduced here: Balance and Variety.

We define *Balance* as the frequency of how often a parametric combination appears in relation to others. This can aid in determining combinations that are over or under represented. The simplest combination is pairwise which can be studied with a 2D matrix. If a matrix is constructed that shows all possible combinations of 2 variables, A and B, the matrix size will be *i x j*, where *i, j* are the number of values that A and B can take on, respectively. Then, the matrix cells can be populated with the raw counts $N_{ij}$ of each combination found in the dataset. The raw cell counts need to be normalized by using variable E based on equal representation in set: Total data set size $N_{tot}$ divided by a, b matrix size to get a "Balance Index" $\varphi_{ij}$ for each A, B combination

$$\varphi_{ij} = \frac{N_{ij}}{E_{AB}}, \quad \text{where} \quad E_{AB} = \frac{N_{tot}}{i * j}$$

If $\varphi_{ij} > 1$ the combination is over-represented and if $\varphi_{ij} < 1$, it is under-represented.

Though tedious, this approach can be extended to 3, 4, n-wise combinations by using higher dimensions matrices. Case study in Sec 7d will illustrate this computation in detail.

***Raw counts of combinations of A, B in data set***

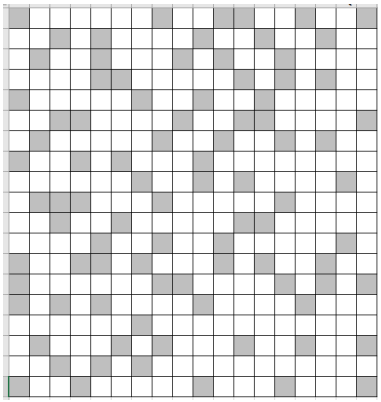
| | | Parameter A | | |
|---|---|---|---|---|
| | | A1 | A2 | A3 |
| Parameter B | B1 | $N_{11}$ | $N_{21}$ | $N_{31}$ |
| | B2 | $N_{12}$ | $N_{22}$ | $N_{32}$ |

***Balance index of combinations of A, B in data set***

| | | Parameter A | | |
|---|---|---|---|---|
| | | A1 | A2 | A3 |
| Parameter B | B1 | $\varphi_{11}$ | $\varphi_{21}$ | $\varphi_{31}$ |
| | B2 | $\varphi_{12}$ | $\varphi_{22}$ | $\varphi_{32}$ |

The other somewhat related aspect of datasets is *Variety*, defined here as the number of variable combinations represented in the data set vs max combinations that could be represented. As with Balance, the quantification below is limited to pairwise (2-param comb) combinations. The matrix shown below for illustrating the idea, shaded cells represent combinations that are present, can be quantified as $a_{ij}$ = 1 vs blanks as 0s.

Populated cells in pairwise matrix
Set $a_{ij}$ = 1 for populated cells



Variety Index V

$$V = \frac{0.5*\left[\sum_{i=1}^{k}\sum_{j=1}^{k} a_{ij}\right]-k}{\left(\sum_{i=1}^{N} L_i\right)^2-\sum_{i=1}^{N} {L_i}^2}$$

Cell populated $a_{ij}$ = 1; empty $a_{ij}$ = 0

**N** = no of factors (parameters)
**L** = levels corresponding to each parameter
Max value V =1

If one desires to include 3, 4, way combinations, etc., one would need to construct n dimensional matrices for balance & variety computation

## 5. Data generation

Before any data is generated, the scope and objectives of the ML study need to be defined. This includes the selection of component/assembly and the performance response(s) of interest. Preliminary studies need to be carried out to determine relevant input parameters and screening studies to reduce input/response variables. Parameter ranges and levels need to be decided on, as well as, CAD geometry creation and CAE settings determined to ensure accuracy, uniformity and validity (geometric and functional). The dataset must be designed (curated) before data is generated. This involves consideration of size, balance, variety, granularity and quality, as discussed in the previous two sections. A typical workflow is depicted in Figure 6. Curation decisions will drive geometry variants that need to be generated, possibly listed as attribute values of variants in a Table. These attributes/parameters are then used in creating the necessary geometry in CAD. If the dataset is being generated only for ML training, the geometry needed for simulation will be created exactly as needed without the need for modification in CAE pre-processor, such as de-featuring, entity merging, mid-surface extraction, etc.

CAD and CAE may not necessarily be from the same vendor suite, so may require a parametric bridge. A decision needs to be made upfront about what results are to be archived, or what granularity is needed for ML; should full detailed nodal/element results be stored, or only some key responses, such as max vM stress, temperature, velocity. The latter could possibly be added to the same Table of variants in order to

associate response with inputs. This also alludes to the importance of data management, which is discussed later.

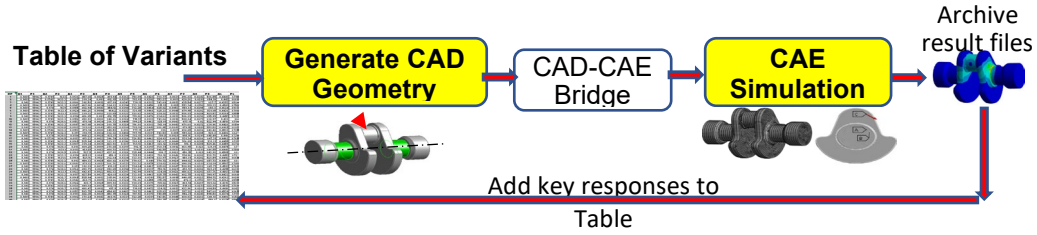



*Figure 6: CAD-CAE pipeline for dataset generation and automation*

The above process assumes manual work. However, ML requires rather large datasets, so the process needs to be automated. Multiple issues arise in doing so and additional tasks need to be undertaken, such as writing scripts and macros. One issue is geometric validity in CAD; how to ensure that parametric feature variations will produce valid geometry, contiguous, connected without feature intersections. Parameter variation ranges need to be carefully determined and geometry must be iso-constrained. These are necessary, but not sufficient, conditions. This is demonstrated through case studies in Sec 6.

## 6. Data structuring & management

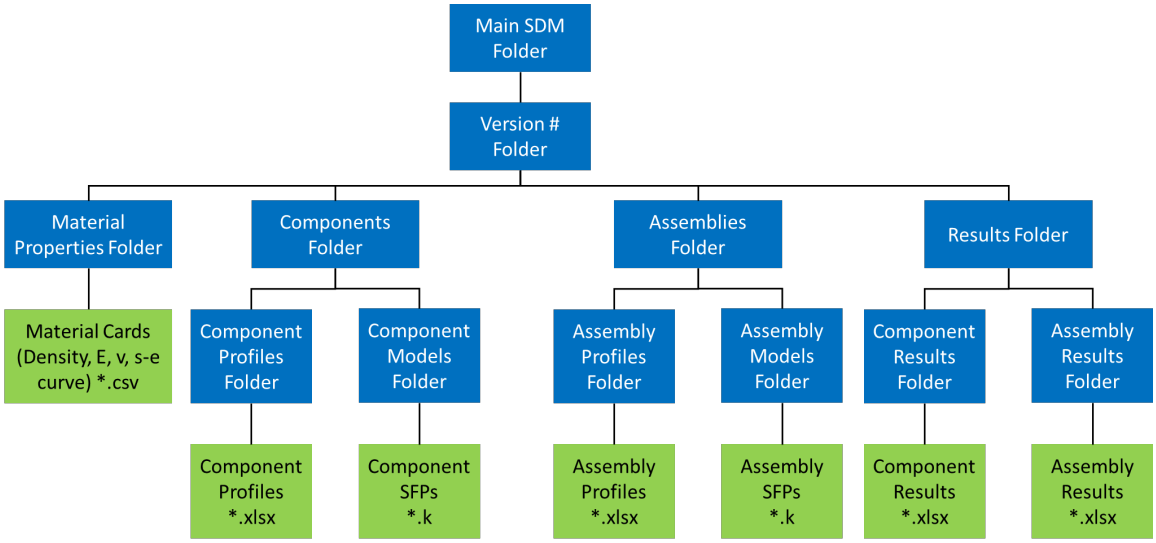



*Figure 7: Data associativity*

Artificially curated datasets for ML typically consist of Variant Attribute Tables, Material properties, CAD geometry models, meshes and CAE results. Each has its own proprietary or standardized format. Since the set may consist of hundreds or thousands of cases, it is important to create a data structure to properly associate these elements to each other. i.e., Case XX has these parameter values, these material properties, this CAD file, these CAE result files, etc. Multi-step simulations, such as assembly simulation based on component simulation, add further requirement for indicating which component simulation results were used in building a given assembly. Case version and configurations need to be managed similar to PDM systems. Figure 7 shows an example of an SDM structure for assemblies. Blue boxes represent folders and green represent data files. As can be seen in the example, there are different file formats in use. In this example, CAD and CAE are in an integrated system, so CAD geometry is not stored in a separate file. The CAD geometry is integrated into the CAE simulation and is generated/updated

automatically based on parameters from the design explorer table). If the dataset is to be published for public use, one needs to convert data files into formats that are more widely usable. For example, STEP/IGES or STL for geometry, CSV for CAE results.

## 7. Selected case studies: Sim data curation & ML

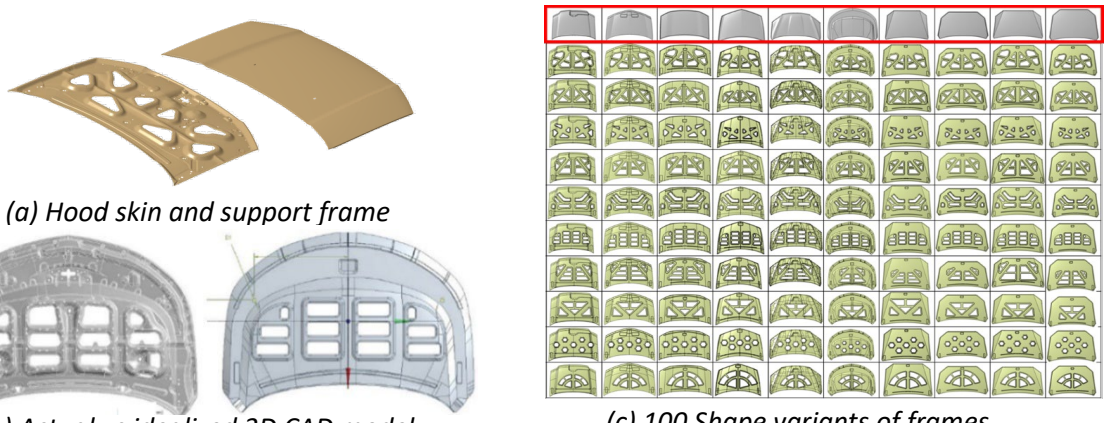
Physics-based simulations are commonly used in analyzing and optimizing product design today. Simulations can predict the structural, NVH, durability, aerodynamics, electromagnetics performance of products among others. However, these simulations require lengthy model preparation and lengthy computations on expensive hardware. Hence design exploration that is required in concept design phases is not as accessible, leaving engineers with few design choices with trial-and-error to pursue rather than finding innovative, optimal designs that can be found through design exploration.

One of the long-asked questions has been whether historical simulation data can be used to train machine learning models to be used for fast design exploration. In some cases, this data is most likely not parametric, nor parametrizable. This is the challenge geometric deep learning solves. With geometric deep learning, we no longer need parametric or parametrizable datasets to train ML models. However, such data may not have variety and balance to be effective, as they were not intended for ML. Hence the need for data curation specifically for ML based design exploration. Selected case studies are presented here.

### a. Automotive Structure Design: Hood Support Frames

#### *i. Dataset Curation*

*CarHoods10k* dataset was generated in a collaborative project between OSU and Honda (HDMA)[31]-[34]. A quick overview is given here. Hood frames are stamped structures to support hood skins (Figure 8a). They are designed to withstand several static and dynamic loads, global and local. The project used 10 reference styling surfaces (skins) of existing hoods and their corresponding support frames to generate 100 shape variants by mapping every frame design to every skin (Figure 8c); for every shape variant (skin-frame combination), 100 size variants were generated by varying feature sizes, to get 10,000 geometries. This "mix-and-match" approach is designed to generate a large amount of geometries that show sufficient variability for ML training. Additional variants can be obtained by varying sheet material and thickness, but does not require any changes in CAD, as it can done in FEA. All 10k models were analyzed in FEA for two static load cases (hood lift and twist). In order to automate CAD generation and FEA simulation, it was necessary to simplify the geometry by identifying and eliminating irrelevant features in the original CAD model (Figure 8b). There is a trade-off between idealization and accuracy.



*(a) Hood skin and support frame*

*(b) Actual vs idealized 3D CAD model*

*(c) 100 Shape variants of frames*

*Figure 8: Hood frame dataset*

Parameter values describing the placement and characteristics of feature patterns were generated using a sampling scheme and stored in a design table [32]. Examples of design variations for the same base surface and features with different parameter values are shown in Figure 9. The CAD-FEA automated, integrated workflow is shown in Figure 10. Macros were used for geometry generation based on Catia PowerCopies (UDF) of features in a given pattern and parametric values in Tables. These are applied to the base component (idealized skin) on which features are created. Construction entities such as offset planes, wire frame geometry, and coordinate systems, which are specific to the CAD process, are created from the base geometry. Initial implementation of this procedure resulted in CAD generation failures largely because of inconsistent combination of feature parameters, resulting in feature interference or non-contiguous geometry. This necessitated the need for a "constraint network" [35]. The constraints in the constraint network are solved sequentially based on the driving parameters, i.e., values in the design table, opposed to driven parameters that are automatically computed based on the driving parameters. CAD generation failures and invalid geometries were reduced from 33 % to 4 %. Thus, in generating large sim datasets, one must consider both simulation and **geometric validity** -another challenge for LLMs.

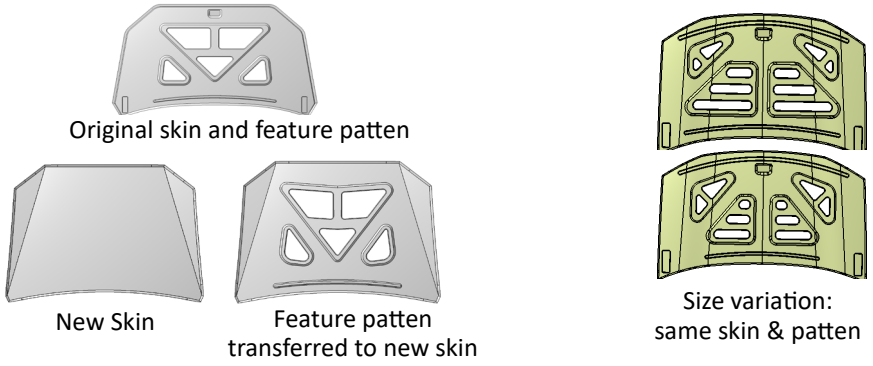



*Figure 9: Conformal mapping of feature patterns*

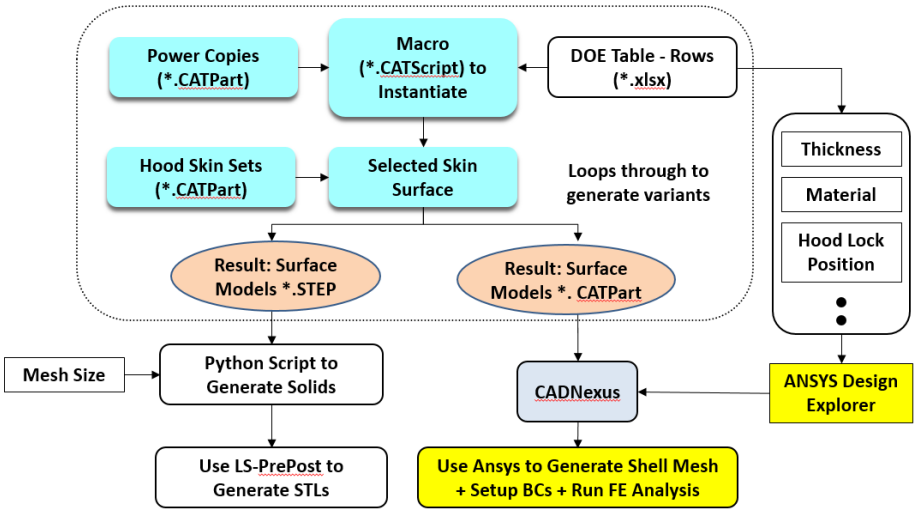



*Figure 10. Automated generation of CarHoods10k dataset*

FEA was performed for hood lift and twist cases. The responses from these load cases were highly correlated [33]. Only maximum equivalent stress [MPa], maximum directional deformation and geometry mass for each design were archived – low granularity output variables. FE setup, mesh BCs were kept consistent in all cases with respect to fixture locations and area, and DoFs constrained with a remote point

that was offset from the frame surface. Prior to automated workflow, the level of idealization was validated by comparing deflections and stresses in idealized geometries to the original reference geometries.

In order to make geometries available to the public that may not have Catia, STL surface meshes were generated via a custom script. All files are watertight and contain properly oriented surface normals.

Such a large and diverse dataset requires a well-defined structure for storage and usage, particularly if it is to be made for public use ( Figure 11); it is organized by skin-frame combinations. CAD data is archived both in Catia format and STL. This dataset has been published for unrestricted use[36]. Multiple parties, internal and external, have used this dataset for training ML. Two of these studies are given below.

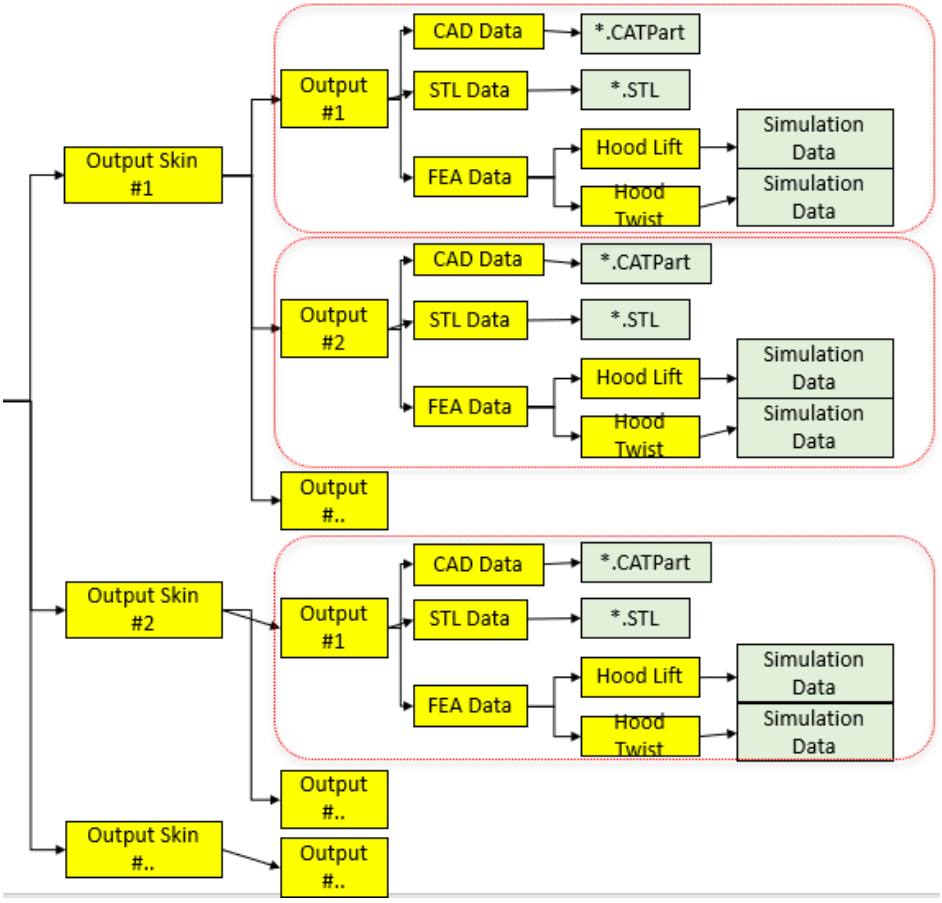



*Figure 11 CarHood10k dataset structure*

*ii. ML investigations with Hood Dataset*

Several ML studies have been undertaken on the hood dataset, both by the publishers of the set and by external parties. A few examples are given here. Altair, now part of Siemens PLM, used CarHoods10k to study the effect of data size and variety on accuracy of ML prediction. They progressively trained geometric deep learning (GDL) predictive models [37] to learn the relationship between maximum equivalent stress and geometry. Starting with a small subset (100 frame designs corresponding to 1 skin, 10 frame feature variants with 10 size variants of each) and progressing to larger (300 designs for 3 skins), and even larger (500 designs for 5 skins), they compared improvements in accuracy versus training data size. After removing outliers, $R^2$ value of 0.87 was achieved in the final study, which was deemed good enough. Figure 12 shows the predicted values for stress with the last of these studies. It can be seen that

an acceptable accuracy was reached and it would have been of marginal benefit from using more data. The amount of data needed obviously depends on the physics and geometry variations. However, when the trained model was tested directly on a 6th skin, the results were poor. But employing transfer learning improved $R^2$ from 0.42 to 0.66. As evident from Figure 8c, top row, skin shapes do not vary quite that much, as opposed to feature patterns of ribs and cutouts. The effect of this difference was illustrated by training based on same frame feature pattern but different skins (Row 7, Figure 8c). It was found that just 100 samples were needed to get $R^2$ value of 0.87.

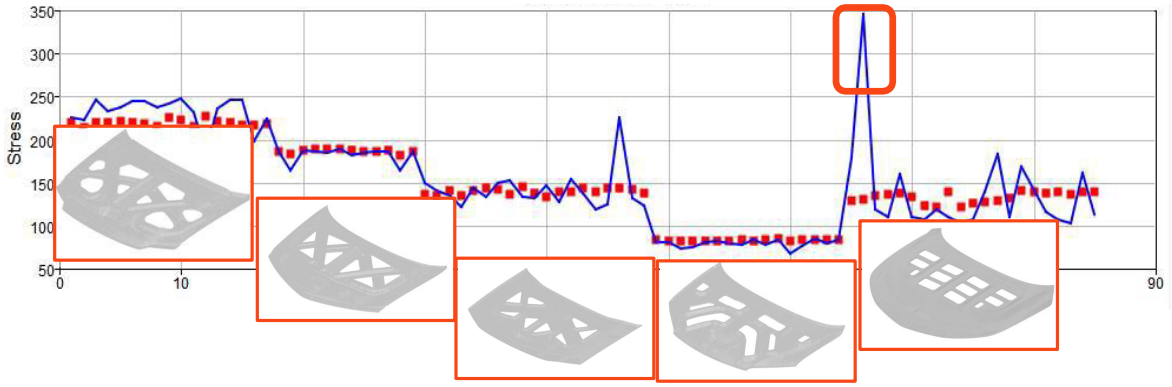



*Figure 12 Performance prediction error*

In a follow up study, Altair PhysicsAI was used with 100 frames in the Skin #2 category. Of the 100 frames 71 were determined to be high quality by the dataset authors. 9 of those frames, representing a geometrically diverse set of designs, were withheld for testing, and the remaining 62 were used to train the model. The GDL model was trained for 14 hours on one NVIDIA RTX A4500 GPU on a laptop. When tested on the 9 test frames (Figure 13), the GDL model achieves an $R^2$ score of 0.79, demonstrating that GDL can learn the relationship between geometry and maximum equivalent stress across a broad range of designs. Perhaps more importantly, the model achieves a Spearman's Rank Coefficient of 0.88, indicating that the predicted design directions (i.e. whether a design change will increase or decrease stress) are well-captured. Each prediction ran in roughly 8 seconds, representing a 13,500x speedup over the 30-hour FEA simulations. The combination of high-fidelity and diverse datasets like *CarHoods10k* and state-of-the-art GDL hold promise for accelerating the product design cycle.

In another study, conducted at ASU [38], a different modality of hood geometries was used, namely pixelated images (3 views: front, top, side). A comparative analysis of three neural network models was conducted: a custom-designed CNN [39], VGG-16 with transfer learning [40], and ResNet-50 with transfer learning [41]. The Custom-CNN architecture had three distinct channels, each dedicated to processing images from one of the three views. Each channel contained three convolutional layers; Each layer used 64 filters with 3×3 kernel sizes, which are small enough to capture detailed features while allowing for a deeper network that can learn complex patterns. The Rectified Linear Unit (ReLU) activation function was used in these layers to introduce non-linearity, helping the network learn complex patterns in the data. ReLU was chosen for its computational efficiency and its ability to mitigate the vanishing gradient problem. Interspersed with the convolutional layers, the Max-Pooling layers (1 to 5) used a 2×2 pooling size to reduce the spatial dimensions of the feature maps, decreasing the amount of computation needed and preventing overfitting.

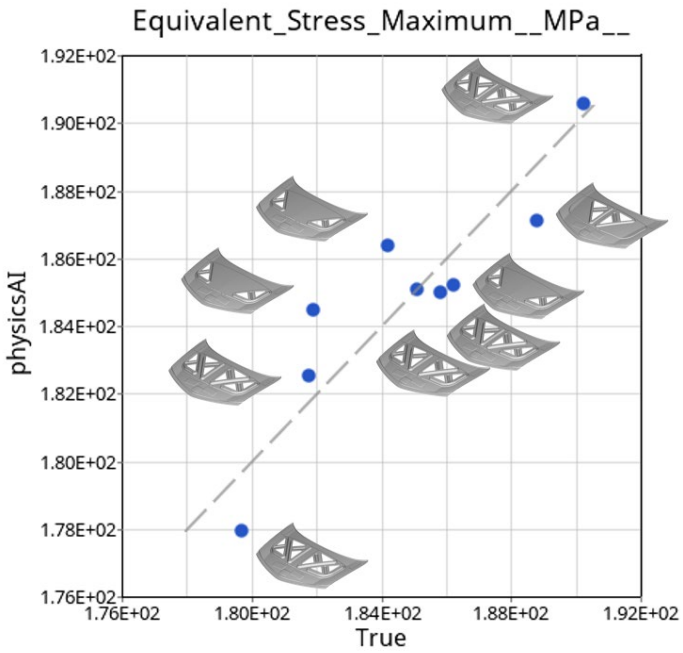



*Figure 13 : GDL to predict equivalent max stress given only the mesh nodes and connectivity. (The dashed line represents perfect predictions (true=predicted).*

The Custom CNN network's predictive accuracy shows a sliding scale of correctness with increasing tolerances. Within a narrow margin of 5%, the network is correct about a quarter of the time. As the margin widens to 10% and then 15%, the accuracy improves to 35.38% and 55.79%, respectively. Finally, with a broad margin of 20%, the network's predictions are 77.22% accurate, indicating that while the network captures the general trend, its precision needs enhancement for tasks requiring high accuracy. VGG16 used 13 convolutional layers, 3x3 convolutional filters exclusively with a stride of 1 pixel, followed by max pooling layers with a 2x2 pixel window, with a stride of 2 pixels the weights of the convolutional layers were frozen as the network was initially trained on a large dataset(ImageNet) efficient in feature extraction. At the end of the network, there are 3 fully connected layers the weights for these layers were retrained. After passing through their respective VGG16 bases, the feature maps were pooled (using global average pooling) and concatenated. The last fully connected layer was tailored for regression and output the maximum stress in the frame. VGG16 led to improved accuracy across all specified tolerance levels compared to the Custom CNN network. Specifically, accuracy within a strict ±5% margin increased to 32.16%, suggesting that the VGG16-based model provided more precise predictions for nearly a third of the dataset. Just like VGG16, using ResNet-50 with transfer learning for each channel involves leveraging a pre-trained network to harness its deep layers for feature extraction. ResNet-50, with its 50 layers, includes a series of residual blocks that make it adept at training very deep networks efficiently. The defining feature of ResNet-50 is its skip connections, which allow gradients to flow through the network directly, combating the vanishing gradient problem often encountered in deeper networks. The application of ResNet-50 has notably enhanced prediction accuracy within various margins of error compared to the previously discussed two methods. Most notably, within the broadest margin of ±20%, the model achieves over 90% accuracy, demonstrating a strong overall predictive capability with the ResNet-50 architecture.

A controlled evaluation of ResNet-50 and VGG-16 on images for the CarHoods10k dataset found that ResNet-50 with residual transfer learning offered the best balance of convergence speed and accuracy. Its frozen backbone leverages ImageNet features, while partial unfreezing adapts high-level features for specific tasks. VGG-16, though slower due to a lack of skip connections, remains useful for simpler tasks or memory-constrained scenarios. A custom CNN trained from scratch underperformed without pretrained weights, but could potentially excel in data-rich domains. Lightweight residual models also nearly match full-size versions for real-time inference, emphasizing the flexibility of residual architectures. In summary, residual architectures, particularly ResNet-50, provide the most versatile solution for a wide range of deployment scenarios, balancing performance and resource efficiency.

*Table 3: Comparison of ML accuracy*

| Tolerance Level | Custom CNN | VGG16 | ResNet-50 |
|---|---|---|---|
| ±5% | 24.26% | 32.16% | 45.54% |
| ±10% | 35.38% | 42.24% | 62.43% |
| ±15% | 55.79% | 65.79% | 74.52% |
| ±20% | 77.22% | 82.52% | 90.08% |

b. **Learning low-dimensional representations from large design data sets**

The following subsections present a set of works where GDL methods, specifically 3D point cloud (variational) autoencoders (V)AEs, have been applied in engineering applications for learning low-dimensional representations from large design data sets in an unsupervised fashion. Also, how the data characteristics were considered in the development of the (V)AE architecture and limitations of the considered approaches are discussed. For illustration purposes, results on the performance of (V)AEs on two data sets: ShapeNet Core (Section 2.b)[3] and CarHoods10k (Section 7.a) [36] are discussed.

As mentioned in Section 2.b, the ShapeNet data set [3] is a repository of 3D designs represented as 3D polygonal meshes (.obj files), images, and metadata. Since the data set comprises the representation of objects from different classes, it allows one to generate subsets with different degrees of diversity of geometric features to train ML algorithms for shape generation and classification. Furthermore, one can resample the geometric data into different modalities, e.g., generating 3D point clouds by sampling mesh vertices, to fit the data to ML architectures, as performed in the following works to train 3D point cloud (variational) autoencoders.

**Data Modality.** Differently from polygonal meshes, 3D point clouds can be represented as lists of coordinates (matrices) with uniform size across a data set without generating geometric artifacts, e.g., missing faces. This property allows the user to develop and train ML architectures for shape generation more efficiently, particularly on graphic processing units (GPUs), since all dimensions of the data set are known. On the other hand, mesh reconstruction for downstream tasks must be addressed in a postprocessing step, which is often challenging when surface normal information is missing, and ML architectures need to cope with the point permutation invariance [42].

**ML Architecture.** One possible way to handle the permutation invariance is to utilize point-wise operators, e.g., 1D convolutions, as utilized in [16],[42]. 1D convolutions process each point individually and yield a matrix of features *F* with dimensions *[N,f]*, where *N* is the number of points and *f* the number of features, which are aggregated by a global operator (in this case, max-pooling) into the latent representation (Figure 14). Hence, the permutation of the points affects the ordering of the rows of the matrix *F*, but not the

latent representation computed by the global operator, which eases the data sampling for training the architecture. Furthermore, encoders with the same structure also allows to "pad" smaller point clouds simply by repeating points, and to process point clouds external to the data set without prior knowledge of the utilized sampling method, which increases the application range of the architecture.

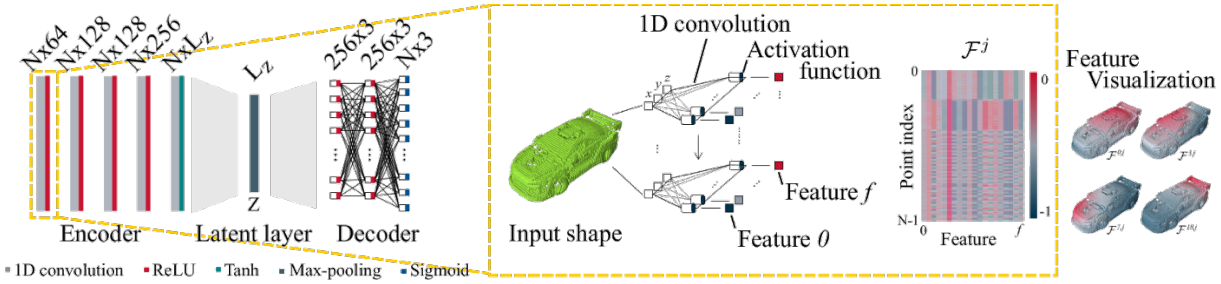



*Figure 14 – Architecture of the autoencoder utilized in [16] with detailed schematic of the point-wise 1D convolution operator, which enables the visualization of features learned in the latent layer (adapted from[43]).*

Additionally to the encoder architecture, the loss function utilized for training the PC-AE is also affected by the permutation invariance of point cloud representations. Differently from point-based shape classification networks [42][44][45], PC-AEs and point cloud variational autoencoders (PC-VAEs) require a loss function that reflects the shape reconstruction quality, which is often computed by measuring the distance between the points in the output point clouds and their corresponding target position. In some works, the authors utilized the Chamfer Distance as (part of the) loss function, which, for each point in the output point cloud, assigns the target coordinate values as the ones from the closest point in the input point cloud [42][46]. The Chamfer Distance function copes with the permutation of the points but requires a search step for matching the points in the input and output point clouds, which can be computationally prohibitive.

Alternatively, if one samples the point clouds according to a consistent ordering, the PC-(V)AE can be trained using a simpler loss function. In [47], is proposed a shrink-wrapping sampling method (SWM), where the point clouds are obtained by shrinking a grid of points onto a target mesh, as wrapping the object with a smooth genus-0 surface (Figure 15a). By sampling the car meshes from ShapeNetCore using the same initial grid, the point clouds in the data set became consistently ordered allowing us to measure the reconstruction error based on the mean-squared distance between corresponding points. Although SWM is computationally more expensive than random vertex sampling, the loss function becomes simpler, which speeds up the training of the PC-AE, without loss of applicability, i.e., the PC-AE can still process unordered point clouds after training with low reconstruction error.

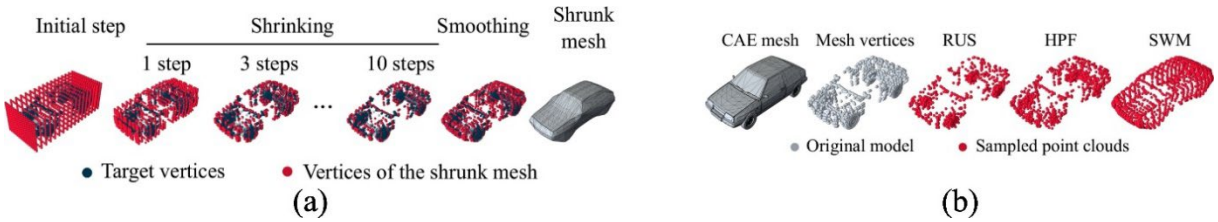



*Figure 15 (a) Example of the shrink-wrapping sampling approach applied to a car shape from ShapeNetCore. (b) Different point clouds obtained for the same shape by directly sampling vertices from a polygonal mesh, randomly sampling vertices with uniform probability*

**Shape-generative and data analysis tasks.** The latent space learned by the PC-(V)AEs is a parametric representation of the input data, which can be utilized for both, generating designs and performing data analysis on the geometries, e.g., aerodynamic drag force prediction by mapping the latent representations to force values obtained through computational fluid dynamics (CFD) simulation with the shapes. Sneha et al. [6][49] performed performance prediction based on latent variables for both aerodynamic optimization and design exploration of car shapes. In [6], the authors also show that the latent space learned by a PC-AE favors the accuracy of the surrogate models used for performance prediction compared to the representations learned by the PC-VAE. Given that both architectures yield similar shape reconstruction quality, the difference in prediction accuracy is possibly caused by the regularization of the latent space, which enforces the similarity of the distribution of samples in the latent space to a multi-variate Gaussian, regardless of the distribution of geometric features in the training data.

A first study to evaluate the capabilities of different machine learning models on this real-world like dataset was conducted by Wollstadt et al.[36]. In a first step, each hood design, given as watertight 3D STL surface mesh, was processed to 3D point cloud format. The x,y,z-coordinates of the point clouds were used to learn a compact low-dimensional representation based on a specified latent space size in an unsupervised fashion using a deep 3D Point Cloud Autoencoder (PC-AE) following Rios et al. [48] and Achlioptas et al. [42]. In the training phase the PC-AE had the task to reconstruct each hood geometry with a minimal reconstruction loss using two different sizes for the latent space vector: L=10 and L=128. Both sizes allow a seamless exploration of the design space. However, as expected, designs created using the higher parameter number (L=128) resulted in less noisy designs and smoother design transitions when changing the latent parameters continuously. As further mentioned in Section 7.a, for each geometry three key results have been calculated and archived: Maximum equivalent stress [MPa], maximum directional deformation [mm] and geometry mass [kg]. To demonstrate the capability of different machine learning models on the hood data set, each geometry was sampled with N=2048 and N=8192 points respectively. For both sample size variations, PC-AEs have been trained using both latent space sizes, i.e. L=10 and L=128. When building prediction models for given engineering data sets, typically the data is split randomly into data subsets, called training and test sets. The training data is used to train the model with a minimum fitting error, while the test data set is reserved for evaluating the performance of the model using unseen data to avoid overfitting to the data and thus ensure generalization capability. If computational resources allow cross-validation it could be a valuable technique to further improve generalization capability. In this study, the data was split randomly into several groups of subsets and used for learning and testing the model. For the *CarHoods10k* data a 90%/10% training/test data set split and a 5-fold cross-validation indicating that model hyperparameters were tuned on 5 different groups of data have been applied. A series of models was trained for each key result (maximum equivalent stress, maximum directional deformation, geometry mass) and then evaluated on prediction quality. The models used were: support vector machines (SVMs), decision trees, linear model, ensemble methods, k-nearest neighbor models, where each model was improved by hyperparameter tuning. The models have been evaluated using standard statistical metrics as mean absolute error (MAE), mean percentage error (MAPE) and $R^2$. In this example, the best-performing models were SVMs with either a polynomial or radial basis function kernel for all combinations of point cloud size and latent space dimension. As one exception, for predicting the directional deformation from L = 10, N = 8192, the Extra-Trees ensemble method performed best. As a result, it has been noted that the models trained on all combinations of L and N achieved satisfying accuracy with a mean percentage error below 10% and below 4% when predicting geometry mass, which is likely due to the more direct relationship between hood geometry and mass compared to

the key features achieved with FEA simulations. A higher latent parameter number also decreased the test errors as a general trend. Based on these models, a typical design optimization pipeline was implemented, which was capable to search optimal input parameter configurations in the latent space of the PC-AE allowing to generate hood geometries with improved design performance. Since the model outputs can be computed fast compared to FEA simulations, it is possible to explore a large portion of a design space with a limited number of function evaluations. Exemplary, differential evolution has been used for a guided optimization to minimize the model outputs for the three hood key factors. In the study, the authors achieved a reduction of the predicted maximum equivalent stress by 45.9%, maximum directional deformation by 7.62% and geometry mass by 8.65% on average. To ensure realism, the resulting designs were visually inspected so that the point clouds reconstructed from the optimized latent representations led to meaningful hood frames.

The PC-AE and GDL frameworks utilize the 3D point cloud data (for car hood geometries) to optimize and predict performance metrics. This approach excels in capturing high-dimensional structural nuances, resulting in robust and accurate predictions. However, it depends on access to simulation-derived point cloud data, which could be computationally expensive to generate. In contrast, the CNN-based image processing model uses 2D images of the hood frames to predict the performance metrics. While this method is straightforward and easily scalable, especially in the conceptual design phases, its accuracy is limited by the lower spatial and geometric fidelity of image data compared to 3D point clouds.

c. **Aircraft Design: Radar Cross section (RCS) prediction**

Radar Cross Section (RCS) is a measure that describes an object in terms of its scattering properties when exposed to incident electromagnetic fields. RCS is directly related to the detectability of an object by radar and depends on electrical object size, object shape and material properties. RCS Evaluation of electrically large objects like aircrafts can be very time consuming. It therefore makes sense to develop ML prediction models based on existing simulation data available from past studies.

Mäuer conducted an RCS study [54] for an aircraft model with different wing sizes and shapes. Parametrized models for a generic twin-engine aircraft were generated for training, testing, and validating the ML models. 4 parameters (wingspan, chord length, sweep angle, dihedral angle) were used with 3 value settings each. The combination of parameters (3×3×3×3) yields 81 corner configurations. To improve coverage, an additional 65 intermediate samples via Latin-hypercube sampling were generated, for a total of 146 unique wing geometries. Each model was meshed in Altair Feko with a maximum element edge of 0.02 λ (≈ 0.17 m at 350 MHz), giving ~200 000 triangular facets per case. Simulations ran for horizontal polarization at $\phi = 0°$ over $\theta = 0°…180°$ in 0.5° steps (361 points), exporting mesh (.cfm) and far-field (.ffe) data into .h3d files. A subset of data is shown in Figure 16. The simulations were performed with Altair *Feko* [50] and the ML prediction model was built in Altair *physicsAI* [37] using geometric deep learning. PhysicsAI uses nonparametric existing data so it can predict directly on the CAD model for real time exploration, followed by validation.

The geometric deep learning approach is designed to learn the correlation between the aircraft mesh and the RCS result curve. Therefore, *physicsAI* needs to process the mesh data and the result data. Feko stores the mesh data in the CADFEKO model file (*.cfm) and the result data in the far-field export file (*ffe). Both data sets are combined in the *.h3d file for each simulation run [50].

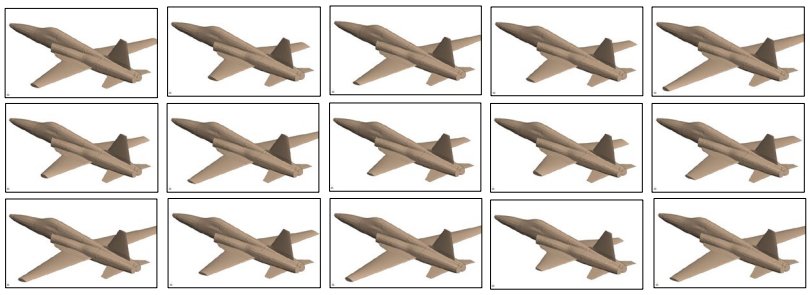


*Figure 16: Subset of the designs for RCS study[54]*

For machine learning, each mesh was parsed into a graph (vertices + edges), where the coordinates of the vertices were normalized to [–1,1], and the RCS values were scaled to dB relative to 1 m². The dataset was split into 131 training and 15 test sets (stratified across sweep and dihedral values). Two geometric-deep-learning (GDL) architectures — MeshCNN [53] and Six-layer Graph Convolutional Network (GCN) with message-passing filters (hidden size 128, ReLU activations) — were compared. On the test set, the GCN achieved an MAE ranging from 0.32 dB (25 m span, 2.5 m chord, 20° sweep, 5° dihedral) to 0.67 dB (30 m span, 3 m chord, 25° sweep, 10° dihedral), with an $R^2 = 0.98$. It was found that the GCN was 15% more accurate and 20% faster to train than MeshCNN. It was trained on an NVIDIA Tesla V100 (16 GB) and it took about 2hours to train.

The error from the GCN can be visualized by plotting the RCS curve for the prediction (red curve) and the simulation result (blue curve) in one Cartesian graph, as shown in Figure 17. Both comparisons, for the smallest and largest error on test data, show high prediction accuracy.

The key lessons and takeaways from this study are:

- **Sampling strategy:** Latin-hypercube sampling dramatically reduced variance in test errors by better covering the 4D parameter space.
- **Mesh normalization:** Scaling mesh coordinates stabilized training and improved convergence.
- **Regularization:** Dropout and weight decay prevented overfitting on this modestly sized dataset.
- **Boundary effects:** Geometries near parameter extremes showed higher errors, indicating the need for additional boundary-zone samples.

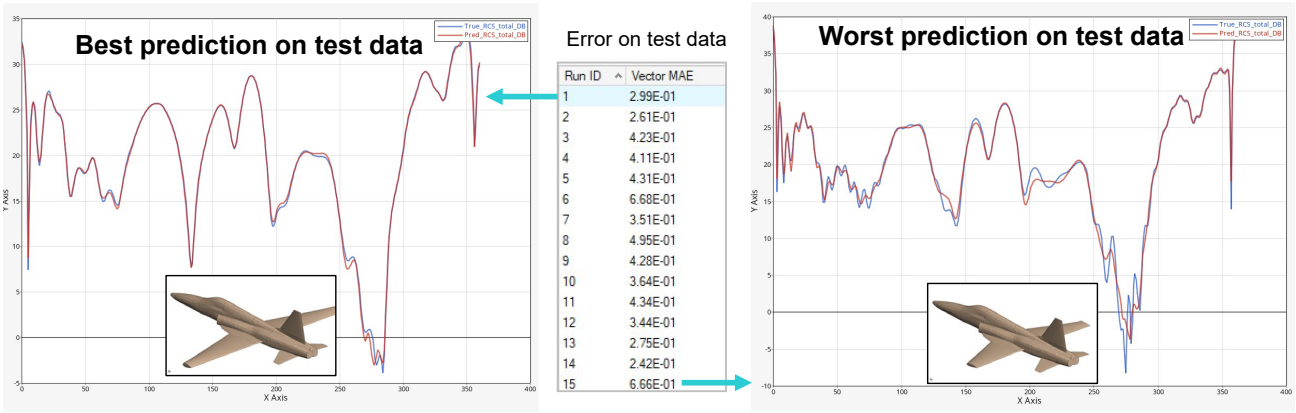



*Figure 17 RCS predicted (blue curve) vs true (red)*

d.**Multi-step Manufacturing Dataset**

This case study was selected because it involves multi-step linked simulations and also non-linearity in analysis. Upper body frames of cars are fabricated sheet metal assemblies manufactured by welding stamped components to form hollow structures with high stiffness to weight ratios. Stamping simulation is nonlinear in every respect: material NL (plasticity), geometric NL (large strains, deformations) and contact (boundary NL). The dataset was curated specifically for ML using a representative body component, namely a stamped and welded T-joint (Figure 18). Manufacturing such fabricated components involves multiple steps; component stamping, vertical and horizontal sub-assembly (joining) and T-joint final assembly/joining.

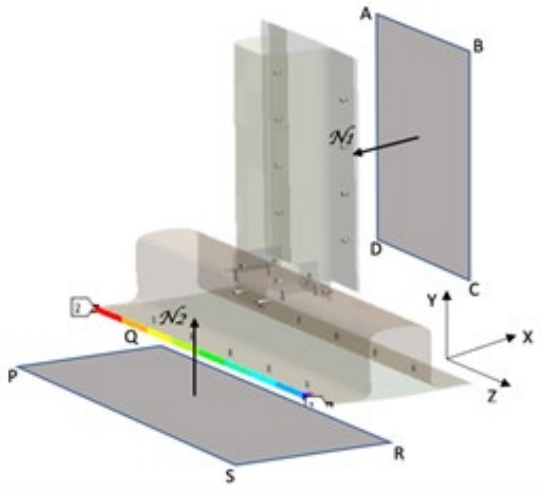



*Figure 18 : T-joint main components*

Data Generation: Three distinct but linked stages of explicit finite element analysis were needed. The first analysis stage, simulates the forming and springback of the vertical and horizontal components from blank sheets using the NUMISHEET 1993 2D U-draw/bending benchmark [29] for its tooling geometry and forming process, as well as, material properties. Parametric variants of components are created in the first stage by adjusting the tooling shape/size, process variables, and materials. Blanks were DP590 1.2mm sheets. Shell quad mesh was used in WB/Dyna simulations. Material model set to isotropic elasticity, VM yield criterion, multilinear isotropic hardening.

The second analysis stage, involves the clamping and joining process, where a vertical formed component and a compatible horizontal formed component are chosen and initially clamped together with flat sheets and brackets. The deformed mesh related stresses and strains are imported from the first stage and transformed into the clamping arrangement because this simulation stage needs inputs (simulation results) from two different forming simulations. To produce different assembly configurations, the pair of vertical and horizontal components are varied while ensuring their compatibility. At the end of the clamping simulation, the deformed meshes and the associated stresses and strains are carried over to the joining simulation. In this stage, spot welds are added between the flat sheet and the flanges of the vertical/horizontal components. Additionally, similar spot welds are applied in between the brackets and components. The clamps are released after the spot welds are in place to allow the assembly to deform in accordance with the residual stresses that were carried over from the forming and clamping simulations.

Data curation must consider variety and balance. All parametric combinations yielded 1440 unique geometries for vertical component and 288 unique geometries for horizontal component. Parametric combinations (full factorial) of assemblies would have require 41,472 clamping and joining simulations

taking 90 minutes each, so not feasible. Instead, random sampling was employed with assigned probabilities; 50 % chance for weld pattern 1 or 2; Random selection of top and bottom components from DOE tables and checking for compatibility conditions. 1000 assemblies were simulated with no duplicates.

The FE simulation results were used to extract stress components and directional deformations along pre-defined paths and edges, which are then converted into parameters that characterizes post-forming springback and post-joining twist. Figure 19 shows examples of results extracted from the stamped components. These results are arranged into a dataset that links the input and output parameters of the workflow. Sensitivity analysis identified most relevant input parameters as material model, blank holding force and certain geometric parameters. Since this project involved many types of data, data formats and several stages of simulation, it was essential to create a data structure to associate them, both for internal use and for public dissemination. The content and organization is shown in Figure 20.

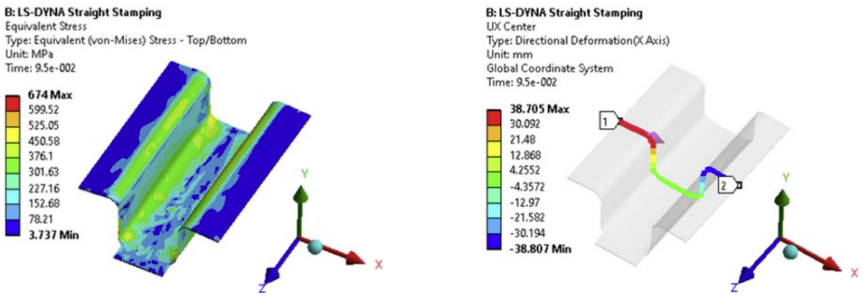



*Figure 19 von Mises stress field (left); cross-section directional deformation (right)*

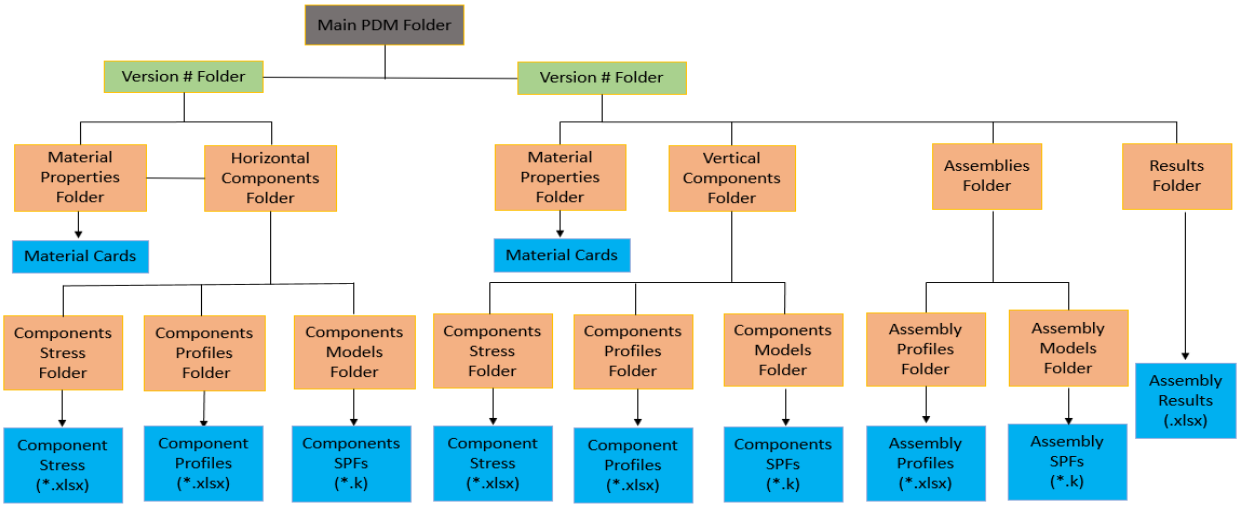



*Figure 20 : Fabricated stamped assembly dataset*

Variety Metrics: To assess variety and balance, the pairwise combination method and indices given in Sec 4 were used. A color coded Balance matrix is shown in Figure 21 to quickly assess under and over representation. Reds indicate over representation, the darker, the higher; blues represent under; whites are equal (near 1). The global statistics for all the parameters in the above matrix are:

$$Min\ \varphi_{ij} = 0.6;\ Max\ \varphi_{ij} = 2.6;\ Mean\ \varphi_{ij} = 1.0;\ Std.\ Deviation = 0.146;$$

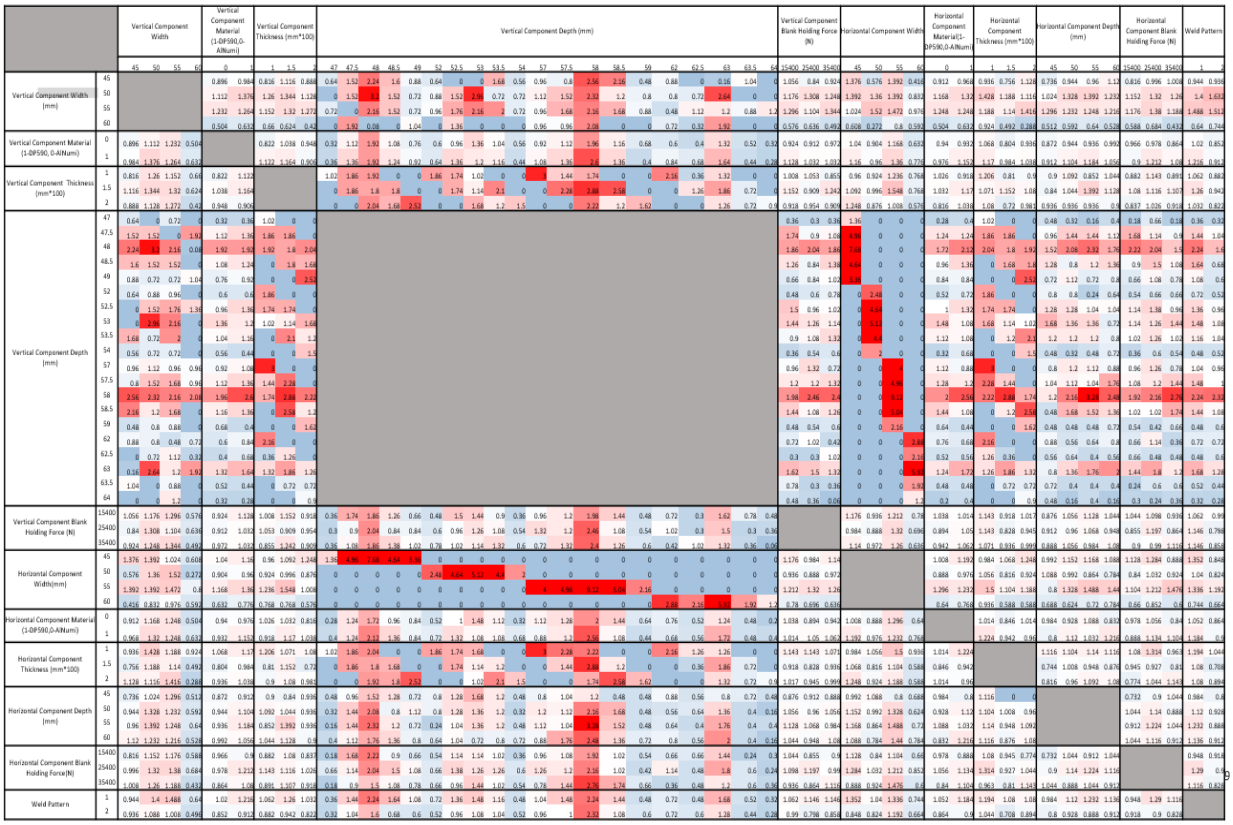


*Figure 21: Pairwise Balance Matrix*

Granularity: ML was considered at two levels: key characteristics of deformed shapes (low fidelity) and full deflection, stress fields (high fidelity). Key parameters extracted for the former were as follows:

Components: 3 deformed (post springback) cross-sections, ends and middle; deformed edges of the hat-section flanges, and all four edges of the unstamped 'flat' components.

Assemblies: unconstrained planar minimum zone magnitudes; angles between fitted planes for the zone fit; twist angles between two nearly parallel edges of an assembly.

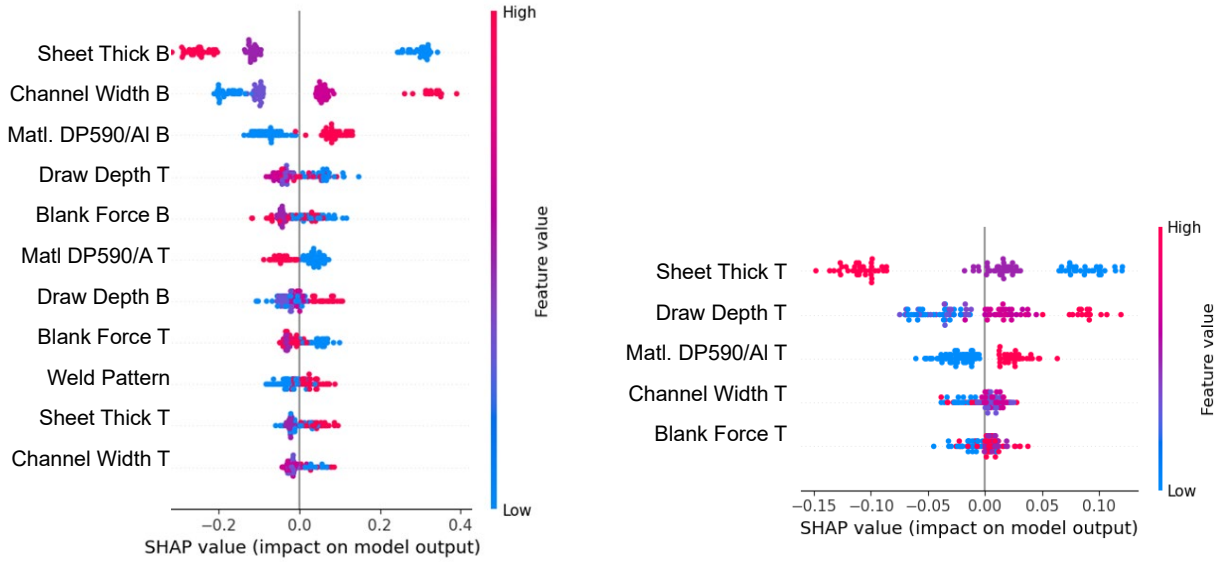
For the unconstrained form zones, all the points for the flat sheets were used, and for the hat sections, points along the flanges were used to construct convex hulls which were then used to determine the minimum normal distances for the zones. To estimate the angles between the planes, a plane was fitted to the points from each of the stamped components of the assembly. The planes were then projected onto the global XZ or YZ plane, following which the angles were computed by the difference in the plane normal and the Y- or X-axis. In order to calculate the final positions of each node along each path, deformations in the x, y, and z directions were generated for each path determined before the simulation's execution. These deformations were then extracted into spreadsheets. Additionally, the equivalent stress and maximum principal stress component in the top, middle and bottom layers were also extracted into spreadsheets.

ML Training: Having high level fit parameters, as well as, full deformation field in the dataset, gives two options for ML, low and high fidelity. ML models were trained to predict 2D stamped component shapes after springback and stress distributions across these shapes. For the assembly dataset, parameters such as unconstrained planar minimum zone magnitudes, angles between component planes, and twist angles

were predicted using machine learning models. ML models explored included linear and polynomial regression, decision trees, gradient boosting regression, support vector regression, and FNN.

In order to move beyond aggregate performance metrics and identify the specific geometric parameters that significantly influence the predictive models, SHapley Additive exPlanations (SHAP) are added to the neural network model used in the T-joint assembly data. The SHAP summary plots, as shown in Figure 22.a, demonstrate that, for zone-size prediction, the bottom-sheet thickness (B) and channel width (B) are the predominant factors driving feature importance, while draw depths and blank-holding forces exhibit minimal influence, as indicated by their near-zero contributions. In the case of twist-angle prediction, as shown in Figure 22.b, SHAP highlights the top-sheet thickness (T) and draw depth (T) as the primary determinants, with material type playing a modulating role—DP590 reduces twist while Aluminum amplifies it. Conversely, channel width and holding forces show negligible effects on the outcome.

SHAP not only serves to validate the quality and effectiveness of our data but also establishes a clear connection between curation decisions and model insights, ensuring that future datasets are both streamlined and highly informative.




*(a)* *(b)*

*Figure 22: SHAP plots for (a) zone size (b) twist angle predictions*

e. **Thermo-fluid Dataset: HVAC design**

This dataset contains seven simulations of an automotive air duct manifold. This system (Figure 23) helps direct the warm and cool air into the passenger cabin, regulating temperature for both drivers and passengers. The computational fluid dynamics simulation provides the pressure distribution throughout the manifold's volume, from the inlet to the outlets into the cabin. Each design is subject to the same input mass flux at the inlet. The seven geometries themselves do not share a common parametric basis and PhysicsAI learns from the shapes themselves without the need for parametric descriptions. Design variations within the set include altered duct lengths, introduced bends, and even a topological alteration from threes outlet to four.

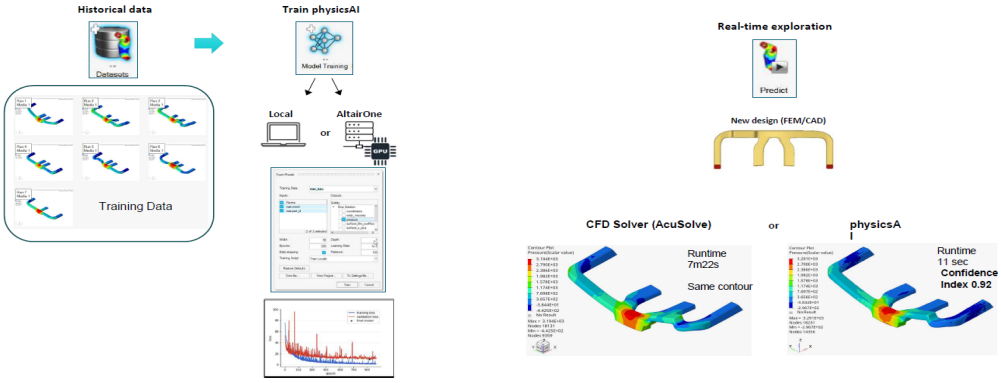



*Figure 23 HVAC system*

The exceedingly limited size of this dataset is typical of many realistic industrial datasets. In contrast to bespoke large public sharable datasets created for benchmarking, companies engaged in design and manufacturing have comparatively limited historical data. One truism is that inferences should be made on data from the same distribution as the training set, so when data is scarce it is unrealistic to expect ML models to broadly generalize beyond the training data. Consequently, the resulting HVAC predictor is limited to modest shape variations like those in the training data. Regardless, this example illustrates that models can be trained successfully with limited data and highlights the need for novelty detection, at inference, to identify shapes that are out of distribution.

## 8. Discussion, Lessons learned, framework for data curation

This paper's primary message is: "Lets' talk about data!". While the majority of today's ML publications focus on algorithms, this paper suggests that researchers give some serious consideration to the characteristics of the data being used in ML. Three critical issues related to simulated data curation for engineering applications were addressed: generation process, structuring/organizing and evaluation of data quality and efficacy for ML. Historical simulation data, even when available, may not be suitable for ML for various reasons, such as lack of variety or uniformity of simulation setup, mismatch of design requirements. Data sets for public use place additional responsibilities on the publishers, such as clear documentation of the content and data organization and design intent, as well as, data formats that can be used without specific commercial packages.

Geometry is at the heart of any simulated dataset. Images, whether single or multiple related views, do not require any specific geometry processors; same for 3D meshes and STL. Traditional data exchange formats, such as STEP and IGES, may be too verbose for direct use. Based on the state of the art in generative AI for engineering applications, we expect a variety of transformative trends to emerge, e.g., generative AI for synthetic data generation, self-curating data pipelines that respond adaptively with AI feedback, federated and privacy-preserving learning approaches, and emerging data quality standards. Currently available LLM models still lack performance in spatial reasoning and geometric processing tasks but works in the literature already report progress in the code generation for CAD representations, particularly with GPT-4[55]. Similarly, CAD-Llama[56], a framework that leverages large language models (LLMs) for computer-aided design (CAD) and parametric 3D model generation, marks a significant step towards addressing the limitations of current LLMs in handling complex 3D geometry. CAD-Llama outperforms baseline models in text-to-CAD tasks and its ability to generate diverse and accurate CAD models from text prompts showcases the transformative potential of generative AI in synthetic data generation for engineering applications. Furthermore, the framework's emphasis on structured data

representation and adaptive learning follows the idea of self-curating data pipelines that respond to machine learning (ML) feedback, highlighting a promising direction for automated data curation practices in engineering. Further examples in this domain are Query2CAD [57] and a multi-agent system for collaborative design by Ocker et al.[58]. Daareyni et al. [59] expand on the idea by integrating also additional input streams such as sketches, blueprints and voice to lower the barrier for non-domain experts to 3D CAD modelling. In conclusion, as we look to the future of LLMs in CAD, we can expect significant advancements in their ability to generate high-quality CAD designs, code, and parametric sequences, with potential applications in various industries, including architecture, manufacturing, and aerospace. Also, while current LLMs may struggle with spatial reasoning and geometric precision, ongoing research and development are likely to overcome these limitations, enabling the creation of more sophisticated and integrated systems that can transform engineering data curation practices and unlock new possibilities for automated design and innovation.

Experienced CAD users may find it difficult to see clearly the advances in geometry generation with LLMs. All major CAD systems have had scripting languages, macros and user defined features to generate CAD models, without ever referring to them as LLMs. Similarly, standard mechanical components can be generated from “catalogs” of parametric Tables that follow a prototypical shape, such as a gear, spring, fastener, etc. That begs the question: are LLMs are a new flavor for these mechanisms?

A number of use cases were included from different design and manufacturing domains. It was demonstrated that the same dataset can be used at different modalities. The three studies conducted by different parties on Car Hoods have used 3D point clouds (HRI study), 3D meshes with nodes and connectivity (Altair study) and pixelated images (ASU study). As well, one use case demonstrated use of the same dataset at two different granularities (T-joint) for low and high fidelity analysis. That study also discussed the complexities involved when an application requires multiple stage analyses, where output from an earlier stage feeds into input from a later stage.

Large public datasets, some of which were reviewed in Sec 2, are collections of unrelated objects and or unrelated engineering function. They are serving a useful purpose in ML for object recognition but are not suitable for geometry associated with specific design functions and engineering performance. A handful of simulated datasets have become available for engineering applications, such as structural design and aerodynamics, some of which were cited or reviewed in this paper. As well, many CAE software companies are creating products to support AI with simulation, such as Altair *PhysicsAI* [37] and *AnsysAI[51]*, but you still need to bring in your own data.

Simulation data sets for ML are typically of non-trivial size (thousands to tens of thousands). They are generated with commercial CAD and CAE/CFD software packages, which may be from different vendors. The consequences are many: Geometry creation needs to be automated since manual work for large datasets is not practical. In order to store design space with ML, these datasets have parametric variants that must generate without error in commercial CAD (geometric validity). At the same time, CAE must ensure analysis validity (accuracy, uniformity of meshing, BCs and other settings), so response can be considered equivalent. CAD and FEA need to inter-operate parametrically and geometrically requiring the development of digital thread and possibly resolving potential inter-op issues with bridge software (e.g., CADNexus for Catia CAD to Ansys CAE). The data transfer and geometry transformation is even more involved in multi-stage simulation, such as that presented in case study Sec 7e; the output of one

simulation stage is the input the next but component positions need to be in new positions with corresponding mapping of CAE results.

This paper argues for bringing discipline and standards for ML data curation, Simulation data sets must be curated with a variety of consideration; Have a clear purpose: application domain, targeted ML algorithms. Careful selection of appropriate modality, granularity; possibly multi-modal, granular. Use of DOE with fractional factorial experiments seems appropriate, but must consider sample size, balance and variety, the metrics for which were proposed in this paper. The size, balance, and variety of datasets are crucial for building and training effective machine learning models. A larger dataset improves generalization and reduces overfitting, while a balanced dataset prevents bias toward the majority class, ensuring fair performance across all classes. A diverse dataset helps the model learn from a range of real-world scenarios, improving its ability to handle unseen data and making it more robust. Combining all three factors leads to better, more reliable models. Determination of dataset size may require an incremental, iterative process as demonstrated in the case study in 7a. Size issue in high dimensionality engineering apps could also be mitigated by reducing dimensionality up front by screening studies or ML Pre-process. Strategies for reducing dimensionality through latent space, as demonstrated by case studies in Sec 7b are also available. This is essential for training efficacy. Feasibility of using a dataset requires also requires matching granularity, modality. The granularity of data impacts machine learning algorithms by influencing the level of detail available for learning; high granularity offers more specific details but can lead to overfitting, while low granularity may simplify the problem but risks losing critical information.

Data driven design holds great promise but also challenges and hurdles. Generation of simulation datasets requires considerable up-front investment, time and personnel. An important question often posed by organizations is advantages/disadvantages of ML over traditional optimization methods, such as Response Surfaces. It is clear even today that ML can handle extremely large number of input variables, such as CAE field data (Displacements, stresses, etc) which is unthinkable for traditional methods. Still ML algorithms are only good within the physics domain and parameter variants that it is trained on; good at interpolation not extrapolation.

## 9. Acknowledgements

*CarHoods10k* dataset generation was funded by Honda (HDMA) through a research contract to Ohio State. Flex assembly datasets funded by NSF Grants 2029905 and 2030093. HRI-EU conducted studies on low-dimensional representations from large design data sets internally. Altair RCS study was conducted by Christopher Mäurer.

## References

[1] https://cs.nyu.edu/~zhongshi/publication/abc-dataset/
[2] J Reizenstein, R Shapovalov, P Henzler, L Sbordone, P Labatut, D Novotny, “Common Objects in 3D: Large-Scale Learning and Evaluation of Real-life 3D Category Reconstruction”
[3] Chang, Funkhouser, Guibas, Hanrahan, Huang, Li, Savarese, Savva, Song, Hao Su, Shapenet: An information-rich 3d model repository 2015
[4] ImageNet: A large-scale hierarchical image database, https://ieeexplore.ieee.org/document/5206848
[5] Whalen, Eamon, Azariah Beyene, and Caitlin Mueller. "SimJEB: simulated jet engine bracket dataset." In *Computer Graphics Forum*, vol. 40, no. 5, pp. 9-17. 2021.

[6] Sneha Saha, Thiago Rios, Leandro Minku, Bas van Stein, Patricia Wollstadt, Xin Yao, Thomas Bäck, Bernhard Sendhoff, Stefan Menzel, "Exploiting Generative Models for Performance Predictions of 3D Car Designs", IEEE Symposium Series on Computational Intelligence (IEEE SSCI 2021), 2021.
[7] Elena Raponi, Mariusz Bujny, Markus Olhofer, Simonetta Boria, Fabian Duddeck, "Surrogate-assisted Topology Optimization of Mechanical Structures", European Cooperation in Science and Technology (COST) Action Training School 2019, 2019.
[8] Felix Lanfermann, Qiqi Liu, Yaochu Jin, Sebastian Schmitt, "Identification of Energy Management Configuration Concepts from a Set of Pareto-optimal Solutions", Energy Conversion and Management: X, vol. 22, pp. 100576, 2024.
[9] Felix Lanfermann and Sebastian Schmitt, "Concept identification for complex engineering datasets", Advanced Engineering Informatics, 2022.
[10] Iqbal H. Sarker, "Machine Learning: Algorithms, Real-World Applications and Research Directions". *SN COMPUT. SCI.* **2**, 160 (2021). https://doi.org/10.1007/s42979-021-00592-x
[11] Qiqi Liu, Yaochu Jin, Martin Heiderich, Tobias Rodemann, "Surrogate-Assisted Evolutionary Optimization of Expensive Many-objective Irregular Problems", Knowledge-Based Systems, vol. 240, pp. 108197, 2022.
[12] Patrick Hoffmann, Kirill Gorelik and Valentin Ivanov, "Applicability Study of Model-Free Reinforcement Learning Towards an Automated Design Space Exploration Framework," 2023 IEEE Symposium Series on Computational Intelligence (SSCI), Mexico City, Mexico, 2023, pp. 525-532, doi: 10.1109/SSCI52147.2023.10371864.
[13] Fabio Muratore, Michael Gienger, Jan Peters, "Assessing Transferability from Simulation to Reality for Reinforcement Learning", IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020.
[14] Rodrigo Canaan, Xianbo Gao, Youjin Chung, Julian Togelius, Andy Nealen, Stefan Menzel, "Evaluating RL Agents in Hanabi with Unseen Partners", AAAI Workshop on Reinforcement Learning in Games, 2020.
[15] Thiago Rios, Bas van Stein, Thomas Bäck, Bernhard Sendhoff and Stefan Menzel, "Multitask Shape Optimization Using a 3-D Point Cloud Autoencoder as Unified Representation," in IEEE Transactions on Evolutionary Computation, vol. 26, no. 2, pp. 206-217, April 2022, doi: 10.1109/TEVC.2021.3086308
[16] Thiago Rios, Bas van Stein, Stefan Menzel, Thomas Back, Bernhard Sendhoff and Patricia Wollstadt, "Feature Visualization for 3D Point Cloud Autoencoders," 2020 International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 2020, pp. 1-9, doi: 10.1109/IJCNN48605.2020.9207326.
[17]G. M. Weiss and F. Provost, "Learning When Training Data are Costly: The Effect of Class Distribution on Tree Induction," Journal of Artificial Intelligence Research, vol. 19, pp. 315–354, Oct. 2003, doi: 10.1613/jair.1199.
[18] Picard, C, Schiffmann, J, & Ahmed, F. "Dated: Guidelines for Creating Synthetic Datasets for Engineering Design Applications." Proceedings of the ASME 2023 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. Volume 3A: 49th Design Automation Conference (DAC). Boston, Massachusetts, USA. August 20–23, 2023. V03AT03A015. ASME. https://doi.org/10.1115/DETC2023-111609
[19] K. D. D. Willis et al., "Fusion 360 Gallery: A Dataset and Environment for Programmatic CAD Construction from Human Design Sequences," Oct. 2020.
[20] K. D. D. Willis et al., "JoinABLe: Learning Bottom-up Assembly of Parametric CAD Joints," Nov. 2021.
[21] Regenwetter, Lyle, Brent Curry, and Faez Ahmed. Regenwetter, L., Curry, B., and Ahmed, F. (October 29, 2021). "BIKED: A Dataset for Computational Bicycle Design With Machine Learning Benchmarks." ASME. J. Mech. Des. March 2022; 144(3): 031706. https://doi.org/10.1115/1.4052585
[22] Regenwetter, Lyle, Colin Weaver, and Faez Ahmed. "Framed: An Automl Approach for Structural Performance Prediction of Bicycle Frames." Available at SSRN 4132282.

[23] Sosnovik, Ivan, and Ivan Oseledets. "Neural networks for topology optimization." *Russian Journal of Numerical Analysis and Mathematical Modelling* 34, no. 4 (2019): 215-223.
[24] Nie, Zhenguo, Tong Lin, Haoliang Jiang, and Levent Burak Kara. "Topologygan: Topology optimization using generative adversarial networks based on physical fields over the initial domain." *Journal of Mechanical Design* 143, no. 3 (2021): 031715.
[25] Ramnath, Satchit et al. (2022). OSU-Honda automobile hood dataset (CarHoods10k) [Dataset]. Dryad. https://doi.org/10.5061/dryad.2fqz612pt
[26] Chang, Angel X., Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese et al. "Shapenet: An information-rich 3d model repository." *arXiv preprint arXiv:1512.03012* (2015).
[27] Kim, S., Chi, Hg., Hu, X., Huang, Q., Ramani, K. (2020). A Large-Scale Annotated Mechanical Components Benchmark for Classification and Retrieval Tasks with Deep Neural Networks. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, JM. (eds) Computer Vision – ECCV 2020. ECCV 2020. Lecture Notes in Computer Science(), vol 12363. Springer, Cham. https://doi.org/10.1007/978-3-030-58523-5_11
[28] Wu, Zhirong, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. "3d shapenets: A deep representation for volumetric shapes." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1912-1920. 2015.
[29] J.Y. Lee, M-G Lee, F. Barlat (2011), Evaluation of Constitutive Models for Springback Prediction in U-draw/bending of DP and TRIP Steel Sheets AIP Conference Proceedings **1383**, 571 https://doi.org/10.1063/1.3623659
[30] Kocer, F (2024), APPLICATIONS OF ML/AI FOR CAE, Data-Driven Engineering: AI/ML Trends Conference, NAFEMS, February.
[31] S. Ramnath, P. Haghighi, J. H. Kim, D. Detwiler, M. Berry, J. J. Shah, N. Aulig, P. Wollstadt, and S. Menzel, "Automatically generating 60,000 CAD variant for big data applications," in *Proc. ASME 2019 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference (IDETC/CIE 2019)*, Anaheim, CA, USA, 2019.
[32] S. Ramnath, P. Haghighi, J. Ma, J. J. Shah, and D. Detwiler, "Design science meets data science: Curating large design datasets for engineered artifacts," in *Proc. International Design Engineering Technical Conferences and Computers and Information in Engineering Conference (IDETC-CIE 2020)*, vol. 83983. American Society of Mechanical Engineers, 2020, p. V009T09A043.
[33] S. Ramnath, J. Ma, J. J. Shah, and D. Detwiler, "Intelligent design prediction aided by experiment design and machine learning in feature based product development," in *Proc. ASME 2021 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference (IDETC/CIE 2021)*, 2021, pp. 1–11.
[34] Abhishek Bolar: Automation and of a Multi-Stage T-Joint Assembly of Stamped Components and Prediction of Performance Parameters using Machine Learning, MSME Thesis, MAE Dept, Ohio State, 2023.
[35] Yiming Jiang, Automated Generation of CAD Big Data for Geometric Machine Learning, MS Thesis, Ohio State, MAE Dept, 2019.
[36] Patricia Wollstadt, Mariusz Bujny, Satchit Ramnath, Jami Shah, Duane Detwiler and Stefan Menzel, "CarHoods10k: An Industry-Grade Data Set for Representation Learning and Design Optimization in Engineering Applications," in IEEE Transactions on Evolutionary Computation, vol. 26, no. 6, pp. 1221-1235, Dec. 2022, doi: 10.1109/TEVC.2022.3147013
[37] Altair physicsAI, Altair Engineering Inc., https://altair.com/physicsai
[38] Prakash Kumar, A Study on the Extraction of Geometrical Parameters from FlexiMech Components and Assemblies and their Impact on Performance: A Machine Learning Approach, MSME thesis, Arizona State, 2024.

[39]O'shea, Keiron, and Ryan Nash. "An introduction to convolutional neural networks." *arXiv preprint arXiv:1511.08458* (2015)
[40]Tammina, S., 2019. Transfer learning using vgg-16 with deep convolutional neural network for classifying images. International Journal of Scientific and Research Publications (IJSRP), 9(10), pp.143-150.
[41]Sahaai, Madona B., G. R. Jothilakshmi, D. Ravikumar, Raghavendra Prasath, and Saurav Singh. "ResNet-50 based deep neural network using transfer learning for brain tumor classification." In AIP Conference Proceedings, vol. 2463, no. 1. AIP Publishing, 2022.
[42] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas J. Guibas, 'Learning Representations and Generative Models for 3D Point Clouds', in International Conference on Machine Learning, 2017.
[43] Thiago Rios, Learning-based Representations of High-dimensional CAE Models for Automotive Design Optimization, Thesis PhD Computer Science, LIACS, Leiden University, 2022.
[44] R. Q. Charles, H. Su, M. Kaichun and L. J. Guibas, "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp. 77-85, doi: 10.1109/CVPR.2017.16.
[45] H. Zhang et al., "Deep learning-based 3D point cloud classification: A systematic survey and outlook," Displays, vol. 79. Elsevier BV, p. 102456, Sep. 2023. doi: 10.1016/j.displa.2023.102456.
[46] Ainesh Bakshi, Piotr Indyk, Rajesh Jayaram, Sandeep Silwal, and Erik Waingarten. 2024. A near-linear time algorithm for the chamfer distance. Proceedings of the 37th International Conference on Neural Information Processing Systems. Curran Associates Inc., Red Hook, NY, USA, Article 2918, 66833–66844.
[47] Thiago Rios, Bas van Stein, Patricia Wollstadt, Thomas Bäck, Bernhard Sendhoff, Stefan Menzel, "Exploiting Local Geometric Features in Vehicle Design Optimization with 3D Point Cloud Autoencoders", IEEE Congress on Evolutionary Computation 2021, 2021.
[48] T. Rios, B. Sendhoff, S. Menzel, T. Bäck, and B. van Stein, "On the efficiency of a point cloud autoencoder as a geometric representation for shape optimization," in Proc. IEEE Symp. Ser. Comput. Intell. (SSCI 2019), Xiamen, China, 2019, pp. 791–798
[49]S. Saha, L. L. Minku, X. Yao, B. Sendhoff, and S. Menzel, "Exploiting 3D Variational Autoencoders for Interactive Vehicle Design," Proceedings of the Design Society, vol. 2. Cambridge University Press (CUP), pp. 1747–1756, May 2022. doi: 10.1017/pds.2022.177
[50] Altair Feko, Altair Engineering Inc., https://www.altair.com/feko/
[51] AnsysAI: https://developer.ansys.com/docs/simai
[52]Edwards, Kristen, Vaishnavi Addala, and Faez Ahmed. "Design Form and Function Prediction From a Single Image." In Proceedings of the International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, {IDETC-21}, virtual, online, 2021
[53]Hanocka, Rana, Amir Hertz, Noa Fish, Raja Giryes, Shachar Fleishman, and Daniel Cohen-Or. "Meshcnn: a network with an edge." ACM Transactions on Graphics (ToG) 38, no. 4 (2019): 1-12. https://doi.org/10.1145/3306346.3322959
[54]C. Mäurer, A. Mudgal and A. Raja, "RCS Predictions Using Geometric Deep Learning," *2025 19th European Conference on Antennas and Propagation (EuCAP)*, Stockholm, Sweden, 2025, pp. 1-5, doi: 10.23919/EuCAP63536.2025.10999884.
[55] Zhang, L., Le, B., Akhtar, N., Lam, S.-K., & Ngo, T. (2025). Large Language Models for Computer-Aided Design: A Survey. arXiv preprint arXiv: 2505.08137.
[56]Li, J., Ma, W., Li, X., Lou, Y., Zhou, G., & Zhou, X. (2025). CAD-Llama: Leveraging large language models for computer-aided design parametric 3D model generation. In Proceedings of the Computer Vision and Pattern Recognition Conference (pp. 18563-18573).

[57] Badagabettu, A., Yarlagadda, S. S., & Farimani, A. B. (2024). Query2CAD: Generating CAD models using natural language queries. arXiv preprint arXiv:2406.00144.
[58] Ocker, F., Menzel, S., Sadik, A., Rios, T. (2025). From Idea to CAD: A Language Model-Driven Multi-Agent System for Collaborative Design. arXiv preprint arXiv: 2503.04417.
[59] Daareyni, A., Martikkala, A., Mokhtarian, H., & Ituarte, I. F. (2025). Generative AI meets CAD: Enhancing engineering design to manufacturing processes with large language models. The International Journal of Advanced Manufacturing Technology, 1–10.