SAND2022-0390C

MACHINE LEARNING CLASSIFICATION FOR RAPID CAD-TO-SIMULATION

Steven J. Owen Armida J. Carbajal Corey D. Ernst Matthew G. Peterson Timothy M. Shead

Sandia National Laboratories,* Albuquerque, New Mexico, U.S.A. sjowen@sandia.gov*

ABSTRACT

We identify machine learning methods that can rapidly distinguish specific categories of mechanisms within a complex CAD assembly with the objective of appreciably reducing user time in preparing models for analysis. We propose a new random forest model, *CubitEDT* and compare with a similar deep learning approach. Custom geometric features are computed using an embedded 3rd party CAD kernel based on common queries often used in meshing applications. We also demonstrate comparable or improved accuracy over current published deep learning classification procedures.

Keywords: machine learning, mesh generation, CAD part classification, ensembles of decision trees, neural networks

1. INTRODUCTION

Complex assemblies frequently include many common mechanisms such as bolts, screws, springs, bearings and so forth. In practice, analysts will spend extensive time identifying and then transforming each mechanism to prepare for analysis. For example, bolted connections may require specific geometric simplifications, specialized meshing and boundary condition assignment. For assemblies with hundreds of bolts, model preparation can be tedious and often error prone. This work uses machine learning methods to rapidly classify CAD parts into categories of mechanisms. Once classified the analyst is able to preview and apply category-specific solutions to quickly transform them to a simulation-ready form.

Figure 1 illustrates the environment as implemented in Cubit$^{\mathrm{TM}}$[1] where volumes of a CAD assembly are first grouped using our proposed classification procedure in real time. In this example, volumes classified as bolts can be quickly reduced to a simulation-ready form with a single operation that may include automatic defeaturing, meshing and boundary condition assignment. The user may preview the reduced form from a wide variety of options and apply the reduction operation to multiple bolts at the same time. Motivated by specific user-driven use cases, additional reduction operations continue to be developed for other part categories.

The focus of this research note is to identify a machine learning model that can predict specific categories of mechanisms in real time from a set of parts in a complex CAD assembly. Our objective is to facilitate rapid category-specific reduction operations with the aim of appreciably reducing user time in preparing models for analysis.

2. CLASSIFICATION

2.1 Background

ML-based part classification is often used for rapid sorting of mechanisms for industrial manufacturing processes. Lambourne et. al. [2] suggests sorting current part classification models into one of four groups: point cloud, volumetric, image-based and graph-based

*Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. SAND2022-XXXX

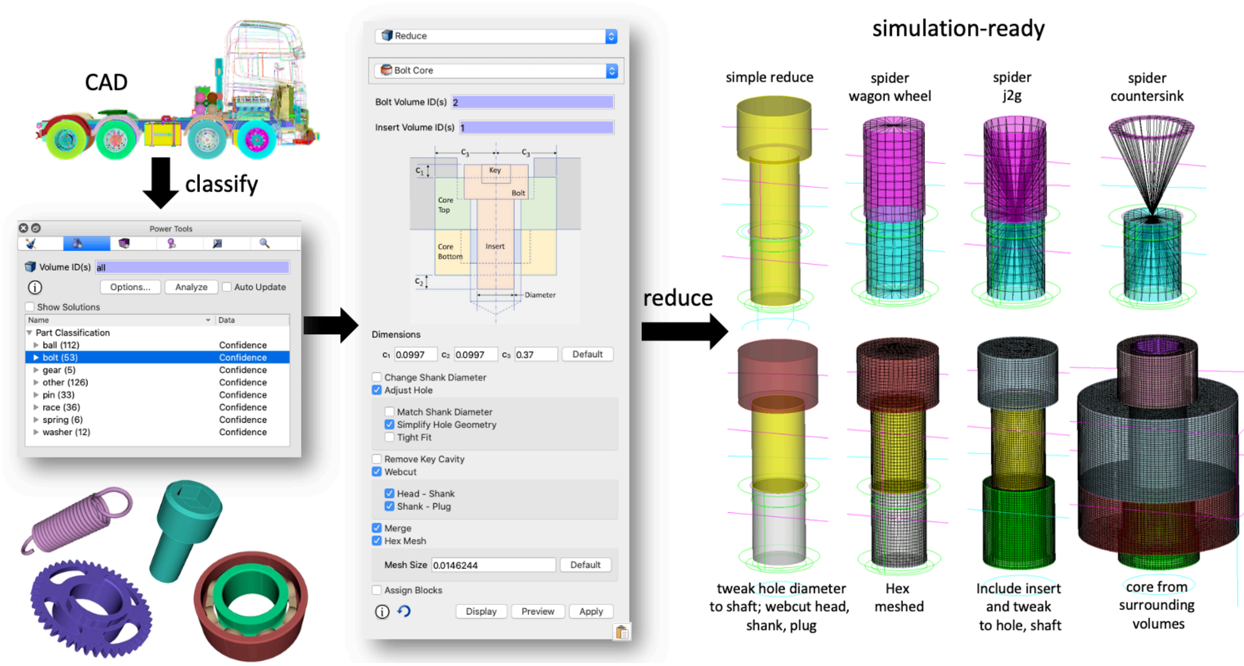

Figure 1: Proposed environment for classification and reduction of fasteners. Each CAD volume is grouped according to a fixed set of categories. In this example, multiple bolt simplification options are presented to the user. They can then rapidly apply similar operations to all bolts in the assembly.

approaches. Reference [2] provides a brief review of each of these methods, citing several examples along with their benefits and drawbacks.

For our application, a complex CAD assembly is usually produced by advanced 3D design tools such as Solidworks [3] or PTC Creo [4] often for the purposes of design and manufacturing. Analysts normally use a modified form of the original CAD assembly as the basis for a computational simulation model. The assembly data consists of multiple parts typically described in a file format such as *.step* or *.sat*. They will describe a hierarchical arrangement of entities including vertices, curves, surfaces and volumes or *boundary representation* (BREP) and where each entity has an underlying numerical description. These formats often use metadata conventions that can identify a name or other attribute which can aid in part classification. However, as we frequently encounter data from numerous sources including legacy CAD assemblies, we cannot assume a consistent metadata convention and must use other means for classification.

We propose a new machine learning approach to part classification based exclusively on derived numerical properties of the BREP. Our method takes advantage of third party CAD evaluation libraries to compute common features such as genus, moments of inertia, volume, tight bounding box, surface areas, etc.

2.2 Features

Training data comes in the form of a fixed length vector, where each value represents a numerical attribute of a geometric volume. We selected 46 features based on common characteristics of curves, surfaces and volumes frequently used for mesh generation. Each feature is easily computed or derived from common query functions of a 3D geometric modeling kernel [5]. Table 1 is a sampling of the features computed for each CAD volume that are used for training data.

We initially noted long training times (i.e. hours or days) with our Neural Network (NN) when using the full 46 features. Our features were highly correlated and determined that feature reduction would reduce the computational time. We reduced the number of features from 46 to 9 features which resulted in a computation time of approximately 9 minutes. Our approach to selecting features was heuristic, there was uncertainty over which of the features would be useful for generating the classification model. This led us to explore pruning the features using Spearman's correlation coefficient [6] which measures monotonic relationships between features. A stepwise removal of features was conducted, eliminating features that had the highest correlation until the remaining features had a Spearman's ρ of 0.29 or less. Additional modifications to the NN parameters has since improved the performance of the 46 and 9 feature NN model computation times (see table 3). Nonetheless, the reduced set of 9 features that resulted from this procedure are shown in table 1 indicated with *.

2.3 Labels

Supervised learning methods for classification also require at least one label associated with each vector of features corresponding to a single volume. For this application, we initially selected 8 labels that reflect common simulation use cases at Sandia Laboratories including: *bolt, pin, washer, nut, spring, gear, ball and race*. One additional category *other*, was used to represent all volumes not identified as one of our initial 8 categories.

Table 1: A small sample of the 46 features computed for each CAD volume used for training data.

Feature	Description
`genus`*	number of through holes
`min_aspect`	tight bbox. min l/w ratio
`max_aspect`	tight bbox. max l/w ratio
`bbox_ratio`*	vol. volume / vol. tight bbox.
`area_vol_ratio`	surface area / volume
`principal_moments`**	ordered moments (3 vals.)
`cylinder_surfs_ratio`	area cylinders / tot. area
`planar_surfs_ratio`	area planar / tot. area
`blend_surfs_ratio`	area blends / tot. area
`reversal_angle_ratio`*	len. crv. w/ext. $\theta > 315°$
`corner_angle_ratio`	len. crv. w/ext. $\theta > 225°$
`side_angle_ratio`*	len. crv. w/ext. $\theta > 135°$
`end_angle_ratio`	len. crv. w/ext. $\theta > 0°$
`area_interior_surfs`*	area surfs w/crv $0° > \theta > 135°$
`len_linear_curvs`*	len. linear crvs. / total len.
`area_high_curvature`*	area surfs. w/high curvature

* indicates features used in reduced set
** smallest and largest values used for reduced features

2.4 Generating training data

To generate training data, a python-based tool was developed that used a 3D CAD kernel to perform evaluations on each volume. While any CAD kernel with the relevant evaluators could be used, we developed our tool using both the Spatial ACIS [5] and Sandia SGM kernels. To evaluate our methods we used 5035 single-part ACIS files labeled manually by displaying an image of the object and selecting from one of the above 9 categories, computing its features and then writing the result to a *.csv* file with its given label.

2.5 Machine Learning methods

We investigated the following classification approaches: ensembles of decision trees (EDT) random forest [7] using Scikit-learn [8] and deep learning technique using neural networks (NN) with PyTorch [9].

Our NN consisted of the scaled set of 46 features as our input layer and 9 nodes in our output layer representing each of our initial 9 classification categories. This sequential linear network contained a hidden layer using a batch size of 128 which doubled in size between Sigmoid Activations to provide the final 9 class output.

An EDT is a collection of individual decision trees, each of which is trained on a subset of the full training data. At evaluation time, the EDTs prediction is a weighted sum of the predictions of each of its individual trees. In prior work, [10][11] the authors used a regression EDT to predict mesh quality based on local geometric features of a CAD model. This work uses a similar approach where we extend EDT to use geometric features for classification.

2.6 Results

We report initial results in table 2 from both NN and EDT models using both the full 46 features and the reduced set of 9 features. To evaluate our results, we use *K-Fold cross validation* [12] using $k = 5$ and $n = 5$, where we assign a randomized 80% for training, and 20% testing over a total of 25 iterations.

Performance of both methods are also reported in table 3 where the total time for training is reported for each of our 4 models. The reported performance in table 3 is the average training time for one occurrence of our K Fold cross validation procedure.

These results show an obvious performance benefit to using EDT over neural networks for our training set, with about a three orders magnitude difference. While both models were above 95% precision and recall when using the full features set, we note a significant degrading of accuracy for reduced features on NN. We observed that although we achieved a small performance improvement for both EDT and NN on our reduced features, there was minimal benefit in pruning features.

2.7 In-situ training and classification

Soon after developing our initial classification methods, it became apparent that our users wanted an interactive method to enhance their training data or add additional custom categories. To accommodate, we have developed a *classify* option within our CAD-based geometry and meshing tool [1] that can dynamically add data to an existing category and/or create additional categories.

In practice, the user may select one or more volumes from within the CAD environment and provide their own label. Features are then computed on the selected volume(s) and exported to a persistent *.csv* file, which will add data to the training set. This then triggers a retraining procedure where all training data is trained

Table 2: Accuracy of EDT and NN models on 5035 CAD parts using 5X5 K fold cross validation.

	EDT				**NN**				
	46 features		9 features		46 features		9 features		
	precision	recall	precision	recall	precision	recall	precision	recall	support
bolt	100.00	98.98	97.46	97.96	98.5	99.0	95	95.6	998
nut	100.00	100.00	100.00	96.15	97.5	86.8	84.0	73.2	114
washer	97.62	97.62	97.44	90.48	94.8	96.2	79.3	76.4	204
spring	100.00	100.00	100.00	91.30	97.2	93.2	89.3	77.6	110
ball	100.00	100.00	100.00	100.00	99.7	100	99.9	100.00	543
race	100.00	100.00	94.29	100.00	95.7	96	90.1	87	148
pin	100.00	100.00	100.00	100.00	98.2	97.8	92.00	94.3	328
gear	100.00	93.33	96.30	86.67	92.0	91.9	79.4	47.2	210
other	98.98	99.79	97.56	98.77	97.8	98.1	89.7	93.9	2380
total	**99.40**	**99.30**	**97.91**	**97.72**	**97.72**	**95.50**	**91.02**	**83.47**	**5035**

Table 3: Performance of EDT and NN models. 5035 models with 5x5 K fold cross validation

EDT		NN	
46 features	9 features	46 features	9 features
0.83s	0.51s	541s	512s

and the model regenerated in real time ready for subsequent queries. This is facilitated by the efficiency of the EDT method described above where the full retraining procedure is usually less than one second on a desktop cpu.

3. COMPARISON

To compare our procedure with other machine learning methods, we use the Mechanical Component Benchmark (MCB) [13][14] which provides two large data sets of over 58,000 mechanical parts. The first set (A) is separated into 68 categories and the second (B) uses a smaller set of about 18,000 objects separated into 25 categories. Each of the objects is in the form of an *.obj* file. We note that the *.obj* format, often used in graphics applications, uses only facets (triangles) to describe the boundary of the object. Although this format is not well-suited to a topology-based approach like ours, we were still able to adapt the data for our EDT classification method.

Faceted formats such as *.obj* do not normally provide explicitly defined topology such as vertices, curves and surfaces. As our features are dependent upon these topological entities, to utilize this data we first generate a *mesh-based* BREP [15], breaking the surfaces and curves where angles exceeded 135 degrees. We also noted that many of the *.obj* parts are comprised of multiple independent shells or volumes where our method assumes a single volume. As well, we also noted many of the objects had gaps or overlaps in their triangle representations which could not be robustly represented using our current methods [15]. As a consequence, we discarded those that did not meet our criteria prior to evaluation.

To facilitate consistency in evaluation, MCB includes separate training and testing collections of parts for both sets A and B. For set A we tested 5713 objects on 68 classes and set B, 2679 objects on 25 classes. We compared our results to multiple published deep learning models reported in, Kim, et. al [2] on the same data sets. We replicate their data in table 4 and add our results as *CubitEDT* for comparison.

Table 4: Comparison of 7 deep learning models to CubitEDT (our model). Replicates data from [2] for *Accuracy over Object* and *Average Precision* for both MCB sets A (68 classes) and B (25 classes) and adds results from our CubitEDT model

	Accuracy (%)		Precision (%)	
Method	A	B	A	B
PointCNN	93.89	93.67	90.13	93.86
PointNet++	87.45	93.91	73.45	91.33
SpiderCNN	93.59	89.31	86.64	82.47
MVCNN	64.67	79.17	77.69	79.82
RotationNet	97.35	94.73	87.58	84.87
DLAN	93.53	91.38	89.80	90.14
VRN	93.53	85.44	85.72	77.36
CubitEDT	**97.04**	**92.9**	**91.79**	**85.81**

We note that CubitEDT accuracy and precision are on par or better than most of the other reported deep learning methods. Kim, et. al [2] does not report performance metrics for comparison.

4. CONCLUSION

We have demonstrated a new machine learning method for classifying mechanical mechanisms in a CAD-based environment that takes advantage of 3D CAD kernel queries. We have shown that our random forest approach (*CubitEDT*) yielded improved accuracy and significantly better performance than a deep learning approach (neural network) and was better suited for in-situ training and retraining. We also identified a minimal set of 9 features that could be quickly computed and used as the basis for training data. Comparison of our EDT approach with other published deep learning approaches yielded comparable or better accuracy.

References

[1] "Cubit Geometry and Meshing Toolkit." `https://cubit.sandia.gov`. Accessed: 2022-01-11

[2] Lambourne J.G., Willis K.D.D., Jayaraman P.K., Sanghi A., Meltzer P., Shayani H. "BRepNet: A topological message passing system for solid models." *CoRR*, vol. abs/2104.00706, 2021. URL `https://arxiv.org/abs/2104.00706`

[3] "MySolidworks." `https://my.solidworks.com`. Accessed: 2022-01-04

[4] "Creo Parametric 3D Modeling Software." `https://www.ptc.com/en/products/creo/parametric`. Accessed: 2022-01-04

[5] "3D Acis Modeler." `https://www.spatial.com/products/3d-acis-modeling`. Accessed: 2022-01-04

[6] Xiao C., Ye J., Esteves R., Rong C. "Using Spearman's correlation coefficients for exploratory data analysis on big dataset: Using Spearman's Correlation Coefficients for Exploratory Data Analysis." *Concurrency and Computation: Practice and Experience*, vol. 28, 12 2015

[7] Breiman L. "Random forests." *Machine learning*, vol. 45, no. 1, 5–32, 2001

[8] Pedregosa F., Varoquaux G., Gramfort A., Michel V., Thirion B., Grisel O., Blondel M., Prettenhofer P., Weiss R., Dubourg V., Vanderplas J., Passos A., Cournapeau D., Brucher M., Perrot M., Duchesnay E. "Scikit-learn: Machine Learning in Python." *Journal of Machine Learning Research*, vol. 12, 2825–2830, 2011

[9] Paszke A. "PyTorch: An Imperative Style, High-Performance Deep Learning Library." pp. 8024–8035. Curran Associates, Inc., 2019. URL `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`

[10] Owen S., Shead T., Martin S. "CAD Defeaturing Using Machine Learning." *28th International Meshing Roundtable, Buffalo NY*, Oct 2019. URL `https://doi.org/10.5281/zenodo.3653426`

[11] Owen S.J., Shead T., Martin S., Carbajal A.J. "Entity Modification of Models.", 9 2020. US Patent: 17/016,543

[12] Brownlee J. "A Gentle Introduction to k-fold Cross-Validation." `https://machinelearningmastery.com/k-fold-cross-validation/`. Accessed: 2022-01-05

[13] Kim S., Chi H.g., Hu X., Huang Q., Ramani K. "A Large-scale Annotated Mechanical Components Benchmark for Classification and Retrieval Tasks with Deep Neural Networks." *Proceedings of 16th European Conference on Computer Vision (ECCV)*. 2020

[14] Kim S., Chi H.g., Hu X., Huang Q., Ramani K. "A Large-Scale Annotated Mechanical Components Benchmark for Classification and Retrieval Tasks with Deep Neural Networks." A. Vedaldi, H. Bischof, T. Brox, J.M. Frahm, editors, *Computer Vision – ECCV 2020*, pp. 175–191. Springer International Publishing, Cham, 2020

[15] Owen S., White D. "Mesh-Based Geometry: A Systematic Approach To Constructing Geometry From A Finite Element Mesh."