A HIERARCHICAL MACHINE LEARNING WORKFLOW FOR OBJECT DETECTION OF ENGINEERING COMPONENTS

Kent, Lee;
Snider, Chris;
Gopsill, James;
Goudswaard, Mark;
Kukreja, Aman;
Hick, Ben

University of Bristol

ABSTRACT

Machine Learning (ML) techniques are showing increasing use and value in the engineering sector. Object Detection methods, by which an ML system identifies objects from an image presented to it, have demonstrated promise for search and retrieval and synchronised physical/digital version control, amongst many applications.

However, accuracy of detection often decreases as the number of objects considered by the system increases which, combined with very high training times and computational overhead, makes widespread use infeasible.

This work presents a hierarchical ML workflow that leverages the pre-existing taxonometric structures of engineering components and abundant digital models (CAD) to streamline training and increase accuracy. With a two-layer structure, the approach demonstrates potential to increase accuracy to >90%, with potential time savings of 75% and greatly increased flexibility and expandability.

While further refinement is required to increase robustness of detection and investigate scalability, the approach shows significant promise to increase feasibility of Object Detection techniques in engineering.

Keywords: Machine learning, Artificial intelligence, Object Detection, Product Lifecycle Management (PLM)

Contact:
Snider, Chris
University of Bristol
United Kingdom
chris.snider@bristol.ac.uk

Cite this article: Kent, L., Snider, C., Gopsill, J., Goudswaard, M., Kukreja, A., Hick, B. (2023) ‘A Hierarchical Machine Learning Workflow for Object Detection of Engineering Components’, in *Proceedings of the International Conference on Engineering Design (ICED23),* Bordeaux, France, 24-28 July 2023. DOI:10.1017/pds.2023.21

1 INTRODUCTION

Prototypes are embodied physically and virtually throughout a design process (Kent et al., 2021). Efficient linking between the two is difficult, time-consuming, and costly with the transition between physical and digital forms often requiring manual effort as each is updated independently (Jones et al., 2020). While increasing efficiency of this process would be highly valuable it is complex to automate, the variety of forms that virtual and physical prototypes may take make even creating a correct association between a physical object and its digital counterpart a complex task.

One method to link the physical and virtual is through Machine Learning (ML), which has been widely used for object detection (Zou et al., 2019). Object detection of physical or virtual parts through image-based ML has been used as a basis for retrieving and comparing matching physical/virtual counterparts. Benefits include better model and product control, synchronised physical/virtual versioning, geometry-driven product search algorithms, and automated detection of defects (Real et al., 2021; He et al., 2022). Convolutional Neural Networks (CNN) are a commonly applied ML method that can identify an unknown artefact from a repository of known artefacts, with a quantified statistical measure of confidence (Dhillon and Verma, 2020). Here, the CNN is ‘trained against a large set of known data (parts) to identify distinguishing features, and then uses this knowledge to identify unknown parts when presented to it. Whilst this is something humans can do intuitively, this skill of identifying what something ‘is’ by sight is complex to emulate computationally (Gopsill et al., 2021).

However, accuracy of identification for CNNs typically decreases as the number of objects that it is trained to detect increases. Real et al. (2021) report an investigation into CNNs for search and retrieval of CAD models based on end-users taking photos real-world objects demonstrating an accuracy of 60% for a CNN trained on 100 objects and 40% when trained on 1,000 objects. This significantly limits the utility of ML techniques for detection of parts used in complex machines, where standard components of many different types and geometries are common. The Mechanical Components BenchMark database of standard engineering components lists 58,696 component models from 68 different classes (see Kim et al. (2020)). With such variety in components and the relatively small geometric differences between many, means that reliable detection using ML is currently infeasible.

This paper proposes and explores an alternative approach to identifying Standard Components (SC) using ML. This involves leveraging predefined taxonomies of parts, digital model repositories, and of close geometric alignment between physical SCs and digital CAD counterparts to create a hierarchical ML approach and significantly increase classification accuracy. In so doing, it provides a means to increase feasibility of ML for part identification, leading to increased feasibility of automated search and retrieval, physical/virtual synchronisation, and additional capabilities.

The paper continues by reviewing related work (Section 2) before presenting the proposed workflow in Section 3. It then tests the workflow through an ML study using a Surrogate Model training approach and TensorFlow, applied to 178 parts across three classes of standard components (Section 4). Results are then presented and discussed in Sections 5 and 6.

2 MACHINE LEARNING IN ENGINEERING DESIGN

Machine Learning (ML) has already had considerable impact across engineering. Design is feeling the impact of ML maturing as an industry-ready toolset, with it being applied across several scenarios to support designers in making the products of tomorrow. Applications strut across a breadth of design and development activities, including:

Simulation: Whereby ML may substantially reduce computational footprint (and hence simulation time) through surrogate models that learn from complex simulations, and take over once their confidence in results is sufficiently high. (Gopsill and Hicks, 2023).

Classifying Product Shape: Whereby ML may aid in understanding human perception of geometry (Gopsill et al., 2021), useful in automating and supporting brand definition, detection and protection (i.e. see Burnap et al. (2016); Ranscombe et al. (2012)), investigated the relationship between human perception of shape and ML interpretation, opening the opportunity for emulating.

Product Interactions: Whereby ML supports identification of associations between products (Maturana and Scherer, 2015) via voxel-based geometric approximations.

Automating Technical Activities: Whereby ML supports automated geometric optimisation Goudswaard et al. (2021), toolpath generation (Kukreja et al., 2020), and production monitoring and remanufacturing process planning (He et al., 2022).

https://doi.org/10.1017/pds.2023.21 Published online by Cambridge University Press

2.1 Object detection in ML

A common application of ML techniques lies in object detection (OD) (Zou et al., 2019), whereby the ML system is ’trained’ to recognise and classify objects from a picture. Training occurs through processing of a large set of prelabelled images through which the system ’learns’ features of each image (and hence object) that allow it to distinguish between objects and correctly classify, which it then looks for in new images when presented to it. One area in which OD is of value to engineering is in automating part recognition and search, where the system automatically identifies and retrieves digital models. There are numerous valuable applications of such technology within engineering. With the growth of the ‘prosumption’ society (Hermans, 2015), large online CAD repositories have emerged containing open-source models for individuals to download and print. Here, OD has value in retrieving models for desired products, where a user may photograph a product or part that they would like to fabricate, and the ML system automatically returns CAD models sourced from online repositories. This has been demonstrated by researchers, who trained ML systems on surrogate models of CAD renders to create such search algorithms (Gopsill and Jennings, 2020; Real et al., 2021). Further utility exists for larger engineering firms where it is common to have products that feature similar components and it is often the case that duplicate designs are generated by the different design teams working on those projects. Reducing and/or unifying designs of standard components can have significant benefits for a firm in terms of reducing supply chain complexity, component data and information management, and inventory management. Here, OD techniques may identify similar or duplicate designs automatically by parsing product data management systems, enabling unification.

With specific regard to recognition of standard components, as will be presented as the focus of this work, OD may enable such capabilities as automatic generation of a Bill of Materials (BOMs), and automated stock monitoring, tracking of assembly and disassembly processes. Further, as standard components are present in near-all engineering systems, OD via their forms may allow generalisation of OD techniques whereby the costly training process (which would require repetition for each bespoke component) may be avoided.

However, while of substantial potential benefit, accuracy of classification of objects using OD is often too low to be considered robust in an engineering context, and decreases as the number of objects increases. Further, the time to generate a functional OD system can be prohibitively long, with a need for full regeneration should any updates or extensions be desired.

3 A HIERARCHICAL WORKFLOW FOR DETECTION OF ENGINEERING OBJECTS

This work proposes a hierarchical ML approach that leverages the established taxonomy of engineering standard components to reduce the size of the dataset, and hence aim to increase accuracy. This workflow leverages the existing digital structure of the mechanical sector to streamline training and detection, resulting in increased accuracy.

Counter to many OD applications, the structure and digital data present in the engineering sector provides advantages for the detection of mechanical parts. Firstly, the objects to be detected are tightly defined, precise, and often fully taxonomised. Where OD techniques must often manage high variety or ambiguity of that to be detected (i.e., the class ‘dog’ contains many 1000s of breeds, with each ‘dog’ being distinct), engineering parts often belong to specific and defined types with little geometric ambiguity (i.e., bolt, washer, nut). For object detection this provides unambiguous models and data structures which may be leveraged to support detection methods.

Secondly, the prevalence of geometric models as counterparts to physical objects supports faster and more robust training. Geometric modelling and product data management systems are ubiquitous across engineering industry, with a high proportion of parts, products, and machines having corresponding digital geometries created regularly throughout their development. As a result, where other sectors require the capture of a large number of images of objects of interest to create a training set, the engineering domain may employ pre-existing precise models to generate controlled datasets automatically. This process, known as *Surrogate Modelling* (Zaki et al., 2016; Gopsill and Jennings, 2020), generates an optimised training dataset using rendered images of geometric models. This process has been shown to enable detection accuracies of >90% for small numbers of objects (Gopsill et al., 2021; Real et al., 2021), with flexibility to account for different lighting conditions and materials. Together, these pre-existing taxonometric structures, precise and defined geometries, and

https://doi.org/10.1017/pds.2023.21 Published online by Cambridge University Press

abundant digital models allow refinement of ML workflows and potential for increased accuracy. While previous work has included all objects to be detected in a single set from which all objects are detected (Real et al., 2021), the proposed workflow leverages known taxonomies of engineering components to partition the dataset into subsets of known types. These partitions are then used to train individual CNNs for OD, each with a decreased quantity of object classes. The workflow of the proposed approach is shown in Table 1 and Figure 1.

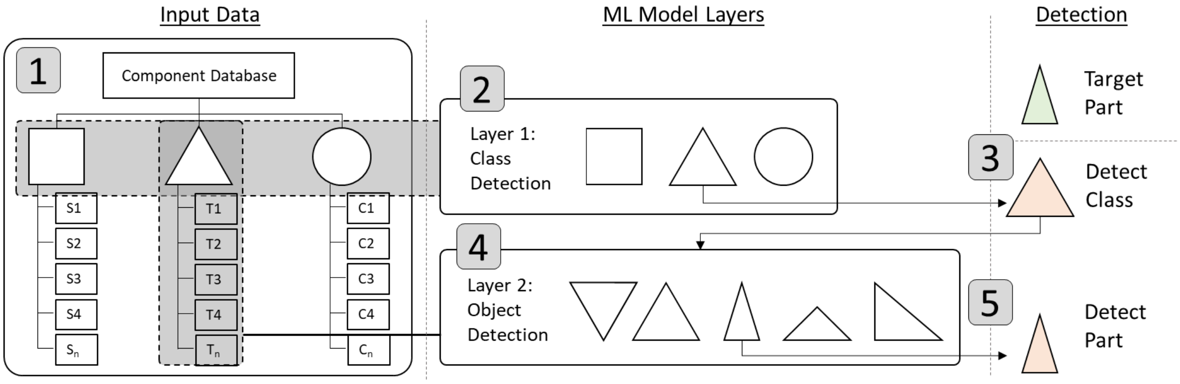


Figure 1. Proposed two-layer hierarchical ML workflow

Table 1. Steps within the hierarchical workflow

Step	**Description**
1: Data Capture	Extract classes of objects and geometry from known taxonomies. Partition dataset according to object classes.
2: Workflow Layer One	Create ML model for classes of interest
3: Detect Class	For target part, detect which class it belongs to
4: Workflow Layer Two	Generate ML model for objects only in class of interest
5: Detect Object	For target part, identify which object it is within class of interest

As noted in other work, reduction of dataset size greatly increases detection accuracy (Real et al., 2021). Using predefined taxonomies of components as a basis (Step 1 in Figure 1), the proposed workflow first trains (Step 2) against defined object classes (e.g., for standard components bolts, washers, collars, hinges). Following detection of the target part as belonging to a specific class (Step 3), a further CNN is trained against only objects belonging to that specific class (Step 4) before final detection in Step 5. Each step in this workflow reduces the required size of the relevant dataset. In Layer One this workflow only identifies the class of object (i.e. not the specific object) using a small set of exemplar components as a training set. In Layer Two, only those objects belonging to the class of interest need be included in the dataset, reducing size of the classified set and potentially increasing accuracy while reducing relative training time.

4 EXPERIMENTAL SETUP

The proposed workflow was tested through a simulation study, trained on 184 objects split over 3 classes of standard component. For reference, expected detection accuracy over this quantity of objects is approx. 60% (see Real et al. (2021). This section details the six steps taken to generate the CNNs and run the simulation.

4.1 Dataset Selection

There are many possible representations of artefacts and approaches to the curation or generation of datasets. Appropriate datasets are time-consuming to create, and several large open databases of CAD repositories are available. This paper uses the Mechanical Components Benchmark (MCB) (Kim et al., 2020), a dataset containing 58,696 models separated into 68 classes. The taxonomy of classes, objects and geometric models that this database provides forms the basis for Step 1 of the workflow. The models are scraped from a variety of sources and each model is in the open .obj format. This dataset

https://doi.org/10.1017/pds.2023.21 Published online by Cambridge University Press

was selected as it is an open dataset, annotated in alignment with the International Classification for Standards (ICS) guidelines published by the International Organization for Standardization (ISO).
To maintain feasible scope for this study, three different classes of object from the database are considered. Each of the classes has roughly the same number of contained models (i.e. versions of that object), but with differing geometric variation between models within a class. Table 2 shows the three selected classes.

Table 2. Selected classes of standard component and models

Class Name	**Geometry Consistency**	**Models [#]**
Slotted Nuts	Small variation in geometry	78
Collars	Medium variation in geometry	52
Hinges	Large variation in geometry	54

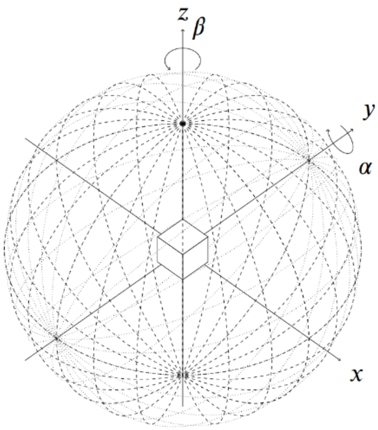


Figure 2. Surrogate model image creation

Table 3. Model used

Layer (type)	**Output Shape**	**Param [#]**
rescaling_1 (Rescaling)	(None, 640, 600, 3)	0
conv2d (Conv2D)	(None, 640, 600, 16)	448
max_pooling2d (MaxPooling2D)	(None, 320, 300, 16)	0
conv2d_1 (Conv2D)	(None, 320, 300, 32)	4640
max_pooling2d_1 (MaxPooling2D)	(None, 160, 150, 32)	0
conv2d_2 (Conv2D)	(None, 160, 150, 64)	18496
max_pooling2d_2 (MaxPooling2D)	(None, 80, 75, 64)	0
flatten (Flatten)	(None, 384000)	0
dense (Dense)	(None, 128)	49152128
dense_1 (Dense)	(None, 54)	6966

00015306_0170_P=-171 Y=-160.png | 00015306_0171_P=-151 Y=-140.png | 00015306_0172_P=-131 Y=-120.png | 00015306_0173_P=-111 Y=-100.png

Figure 3. Generated images during data augmentation step

Of note is that the objects within each class are all standard components, and hence have limited between-component variation compared to many other cases in which object detection is applied. This is for two reasons; with high similarity inherent in many engineering components it is useful if OD can distinguish even between objects that humans may struggle to identify; and second, high similarity presents a difficult challenge for OD to stress-test accuracy and capability. As such, detection accuracy would intuitively be lower here than seen in other works.

4.2 Data augmentation

To overcome the challenge of requiring large manually-generated datasets to train a CNN, a common approach is to perform a data augmentation step. Here this involved extending the dataset using a virtual surrogate model and a series of generated representations of that model for training (see Zaki et al. (2016); Gopsill and Jennings (2020); Real et al. (2021). The Unreal Engine was used to augment the dataset. To generate the images the objects were imported and 640x600 pixel renders were captured at 20 degree increments, rotating around the y and z axis (shown in Figures 3 and 2). The background is a colourless void to neutralise any potential environmental noise. Three point-lights

https://doi.org/10.1017/pds.2023.21 Published online by Cambridge University Press

are used to illuminate the model and a realistic brushed aluminium texture is applied. This led to the creation of 324 images for each object, with a selection shown in Figure 3.

4.3 CNN development

The CNN was developed using Keras and trained using TensorFlow 2.11, using Colab Pro+. The instance is a high RAM shape with 51GB RAM. The dataset is shuffled and split into three sets before being processed (normalised). The split is 70% training data, 20% validation data, and 10% test data previously unseen to the CNN. Table 3 shows the CNN model, the shape and dimensionality. The final row is dependant on the number of models in the class (see Table 2).

4.4 Hyperparameters

A batch size of 9 was used to speed up the learning with 'Adam' optimisation, with a learning rate of 0.001. 10 epochs were used to illustrate the model progress, as the complete model evolution is not required for this study. 3 convolutional layers were used with 'relu' as the activation function, with two dense layers. These hyperparameters were found to balance accuracy of model output and training time using cloud computing capability. Increased number of epochs would be beneficial to increase accuracy, while incurring a time-cost.

4.5 Evaluation

The CNN was evaluated in terms of its effectiveness at recognising the test dataset, with the effectiveness measure achieved algorithmically as part of the model training process. Both accuracy on the training set (i.e., detect a part from the image set used to train) and validation set (i.e., detect a part from images not previously seen by the model) are reported. The key performance indicators are that each of the three models are able to successfully identify the learned classes. For Layer 1 (see Figure 1) of the proposed workflow, this refers to ability to detect the class of object (i.e., hinge, collar, or slotted nut). For Layer 2, this refers to ability to detect the specific collar, hinge, or slotted nut from the objects within the class (i.e., hinge A, hinge B, etc.).

4.6 Simulation cases

The model was trained and evaluated according to two pipelines, see Table 4. The first trained and detected against all 184 objects. The second followed the proposed workflow, training and detecting first against the three class types (Layer 1), and then against the objects within the target class (Layer 2). This required generation of five CNNs: one for the All in one case, one for Layer One of the proposed workflow, and one for each class in Layer Two.

Notably, the CNN for Layer 1 attempts only to identify the type of object rather than the specific object itself. To achieve this the objects of each type were aggregated into a single class, with the model then asked to recognise the class to which an object belonged. I.e. Class one contained all images for all 78 models of the slotted nuts type, with objects recognised just as a slotted nut, rather than the specific object.

5 RESULTS

The model was trained and tested in each case presented in Table 4. This section presents general results for training times, followed by results for each workflow in isolation.

5.1 Training time

Training an ML model is a computationally intensive task, with training time increasing as number of classified objects increases. As all pipelines within this study used identical hardware, training time may be used as a relative measure. Table 5 shows training time for each CNN. Training of the entire proposed workflow (total of 7988s) is approximately equivalent to training of the all-in-one approach (7423s) should all classes require training. However, substantial time may be saved should elements of Layer 2 not be required (i.e. 64% time saving vs. All-in-One to train only Layer One and Collars within Layer Two).

https://doi.org/10.1017/pds.2023.21 Published online by Cambridge University Press

Table 4. Model test cases

Case Pipeline	Description
1: All in one	Trained against all 184 objects, each as its own class. Test attempts to identify specific object from entire set.
2: Proposed Workflow	**Layer 1:** Trained against all 184 objects, but separated into only 3 classes by object type. Test attempts to identify to which class the target object belongs. **Layer 2:** Trained against all objects in target class, Test attempts to identify specific object within the class.

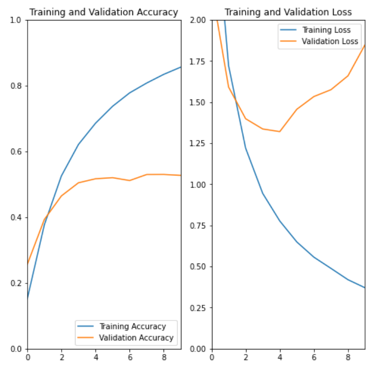
Table 5. Training times and parameters across workflows. Times are per epoch.

Training Case	Objects	Models [#]	Accuracy	Precision	Recall	F1	Training time [s]
All in one	All	184	69.5%	73.3%	72.3%	71.2%	7423
Layer 1	All	184	98.2%	98.3%	98.1%	98.2%	596
Layer 2	Slotted Nuts	78	50.5%	54.8%	52.0%	50.7%	3116
Layer 2	Collars	52	93.0%	94.1%	94.5%	94.2%	2071
Layer 2	Hinges	54	87.7%	88.2%	87.6%	87.6%	2205

Figure 4. Loss and accuracy for training and validation steps for all in one

(a) Hinges

(b) Collars



(c) Slotted Nuts

Figure 5. Loss and accuracy for training and validation steps for layer two

5.2 Case 1: All-in-One

Figure 4 shows test and validation accuracy and loss for the All-in-One, which trained against all 184 objects within a single CNN. Detection accuracy can be seen to be stable at approx. 70% from epoch 6, approximately in line with detection accuracies seen in other works with similar scale of classes (Real et al., 2021). It is notable that the validation loss is higher than the training loss, which typically indicates over-fitting and an inability for the model to generalize to new data. This is perhaps due to

https://doi.org/10.1017/pds.2023.21 Published online by Cambridge University Press

the high similarity between the parts, where the model struggles to identify features in new data that distinguish from existing models and hence creating an over-fit.

5.3 Case 2: Hierarchical workflow

Results for each workflow layer are presented in Table 5 and Figure 5. Looking first at Layer One, it is evident that the CNN has substantially exceeded the accuracy of the all-in-one case (98% vs. 70%). The improvement over the all-in-one is due to the separation of the 184 objects into only 3 classes, with accuracy in line with expected results seen in similar studies (Real et al., 2021).

Looking at Layer Two, in which a CNN was trained for each class of object, it is evident that the object under consideration impacts accuracy of detection. Both the CNNs trained on hinges and on collars achieved detection accuracies of approx. 90%, while the CNN trained on slotted nuts achieved an accuracy of only approx. 50%. This may be due to the higher size of the slotted nut dataset (78 objects vs. 52-54 for other classes), or some feature of the object geometries themselves confounding detection. With high similarity of distinguishing features within geometries (i.e., slots on slotted nuts) it is possible that training is causing overfitting. A further potential cause is the high between-view variation present for each single object (i.e., visual variation for a slotted nut is large across views), requiring the CNN to recognise that despite a lack of consistency two images may be of the same object. Remedying this issue may require the implementation of a multi-view CNN (i.e., see Su et al. (2015)).

However, while further work should investigate how model geometry influences accuracy and may be improved, these results show viability of the process. The substantially higher detection accuracy even between highly similar objects, with models generated in substantially less time, creates potential for higher feasibility of use of CNNs for automatic part detection.

6 DISCUSSION

The discussion considers three perspectives; the performance of the hierarchical workflow, benefits for engineering, and limitations and future work.

Viability of the Proposed Workflow: Results presented in Section 5 show technical viability of the proposed workflow to increase feasibility of CNN-driven object detection for engineering components. In some cases accuracy is increased by the two layer approach by greater than 20%; detection accuracy across layers for collars is 91.3% (98.2% layer one and 93.0% layer two collars), also with a reduction in training time of 64.1% (7423s all-in-one, 2667s layer one + layer two collars). However, they also highlight some of the challenges that creating such CNNs presents, and highlight sources of error.

The case in which the workflow was applied is inherently challenging for CNNs. Standard Components of a single type do not vary substantially from each other and class sizes here are reasonably large - that the CNN was able to achieve accuracy in some cases of >90% despite this challenge shows promise. That training times for the proposed workflow were at worst equivalent to the all-in-one approach with potential to substantially reduce time-cost (by as much as 74% if training for only the first layer (Layer One) and a single class of Layer Two) also suggest feasibility and scalability.

There remain however elements of the workflow that require further investigation. The lower accuracy of the Layer Two CNN trained on slotted nuts (51%) compared to both hinges (88%) and collars (93%) requires further investigation. This may be due to over-fitting due to geometry of parts, an insufficiently small set of representative parts trained for Layer One, or imply the need for a multi-view CNN implementation, and requires further investigation. As such, while results show promise, improvements must be made before the workflow could be considered technically robust. It may be that measures of similarity of a dataset can be created that describe suitability for and ML approach.

Workflow Value for Engineering: A key strength of the proposed workflow lies in its pragmatic, consistent, and step-wise approach to CNN implementation. It enables a train-as-needed approach, where each CNN in Layer Two need only be created when the objects within it become of interest. Further, each CNN may be updated as needed without re-training the whole set should additional objects or classes be added. In typical implementations, the entire CNN must be retrained following every change, incurring the full associated time-cost. An additional value is in the consistent pipeline that the workflow uses, that does not require any reconfiguration to accommodate new classes or objects. The surrogate model approach allows automated training on any digital file (which are abundant in the engineering sector), with the training pipeline then identical for each CNN at each

https://doi.org/10.1017/pds.2023.21 Published online by Cambridge University Press

layer. As such maintenance of the CNNs over time is viable for non-expert parties in industry, where once the pipeline is created, it may be updated and expanded by users with no required configuration.

Limitations and Future Work: Key limitations of the work lie in investigating and improving accuracy in certain areas. The low accuracy of Layer One is unexpected in comparison to the accuracy of the all-on-one workflow, requiring investigation into potential over-fitting, size of training set for Layer One, or interplay between geometry and accuracy. Further, the varying accuracy at Layer Two implies a geometric dependence between some standard components and classification accuracy that should be investigated. Finally, it is apparent that for some CNNs accuracy had not yet plateaued and that further epochs would bring benefits. As the number of required epochs directly impacts training time, the cost/benefit of increasing epoch quantity should be established.

Once such issues are understood, further work should then probe the capabilities of the approach. The extendability of the workflow across classes should be evaluated to identify both number of classes and sizes of class to maintain suitable accuracy. Further, while the CNN has shown capacity to distinguish between very similar standard components, it may struggling with issues of geometric invariance. For example, where an object changes in scale alone (i.e. all features remain consistent relative to one another), there is little on the object for the CNN to discriminate against. Finally, quality of input data is critical for CNN training, and the impact of the readily available databases of lower-poly geometries should be evaluated to determine whether they are sufficient for robust accuracy.

Following such investigations, further work should then investigate implementation in real cases. Here both the training and validation data were 3D renders. While surrogate modelling has shown capability when detecting from images of real components (Gopsill and Jennings, 2020), the proposed workflow should also be validated using this approach. Following, utility to solve real problems should be verified, such as automated Bill of Materials generation, or assembly search and retrieval. Such implementations will require detection of single objects in scenes of several, creating further challenges of occlusion, image segmentation, and nested assemblies.

7 CONCLUSION

This paper has presented and evaluated a two-layer hierarchical workflow for object detection in the context of engineering, using convolutional neural networks (CNNs). While object detection using CNNs has huge potential in a range of engineering applications, the time-cost of training and decreasing accuracy as number of objects increases decrease feasibility of implementation. While further investigation of some results is required, two-layer workflow shows potential to achieve detection accuracy of >90% for datasets of approx. 200 objects (20-30% higher than using other workflows) while also decreasing training time by up to 64%, and allowing streamlined updating and extension without full retraining of the CNN. By leveraging existing taxonomies of components and digital models that are abundant within the engineering sector, this paper then demonstrates potential for feasibility of object detection in the engineering sector to be substantially increased.

ACKNOWLEDGEMENTS

The work reported in this paper was conducted at the University of Bristol in the Design and Manufacturing Futures Lab, and supported by the EPSRC grant refs: EP/W024152/1, EP/V05113X/1, and EP/R032696/1.

REFERENCES

Burnap, A., Hartley, J., Pan, Y., Gonzalez, R. and Papalambros, P.Y. (2016), "Balancing design freedom and brand recognition in the evolution of automotive brand styling", Design Science, Vol. 2, p. e9, https://doi.org/10. 1017/dsj.2016.9.

Dhillon, A. and Verma, G.K. (2020), "Convolutional neural network: a review of models, methodologies and applications to object detection", Progress in Artificial Intelligence, Vol. 9 No. 2, pp. 85–112, https://doi.org/ 10.1007/s13748-019-00203-0.

Gopsill, J., Goudswaard, M., Jones, D. and Hicks, B. (2021), "Capturing mathematical and human perceptions of shape and form through machine learning", Proceedings of the Design Society, Vol. 1, pp. 591–600, https://doi.org/10.1017/pds.2021.59.

Gopsill, J. and Hicks, B. (2023), "A sustainable simulation framework for computational design using web service concepts", Proceedings of the Design Society. In Review.

https://doi.org/10.1017/pds.2023.21 Published online by Cambridge University Press

Gopsill, J. and Jennings, S. (2020), “Democratising design through surrogate model convolutional neural networks of computer aided design repositories”, in: Proceedings of the design society: design conference, Vol. 1, Cambridge University Press, pp. 1285–1294, https://doi.org/10.1017/dsd.2020.93.

Goudswaard, M., Hicks, B. and Nassehi, A. (2021), “The creation of a neural network based capability profile to enable generative design and the manufacture of functional fdm parts”, The International Journal of Advanced Manufacturing Technology, Vol. 113 No. 9, pp. 2951–2968, https://doi.org/10.1007/ s00170-021-06770-8.

He, Y., Ma, W., Li, Y., Hao, C., Wang, Y. and Wang, Y. (2022), “An octree-based two-step method of surface defects detection for remanufacture”, International Journal of Precision Engineering and ManufacturingGreen Technology, pp. 1–16, https://doi.org/10.1007/s40684-022-00433-z.

Hermans, G. (2015), Opening up design: Engaging the layperson in the design of everyday products, Ph.D. thesis, Umeå University.

Jones, D., Snider, C., Nassehi, A., Yon, J. and Hicks, B. (2020), “Characterising the digital twin: A systematic literature review”, CIRP Journal of Manufacturing Science and Technology, Vol. 29, pp. 36–52, https://doi.org/10.1016/j.cirpj.2020.02.002.

Kent, L., Snider, C., Gopsill, J. and Hicks, B. (2021), “Mixed reality in design prototyping: A systematic review”, Design Studies, Vol. 77, p. 101046, https://doi.org/10.1016/j.destud.2021.101046.

Kim, S., Chi, H.g., Hu, X., Huang, Q. and Ramani, K. (2020), “A large-scale annotated mechanical components benchmark for classification and retrieval tasks with deep neural networks”, in: Computer Vision – ECCV 2020, Springer International Publishing, Cham, pp. 175–191.

Kukreja, A., Dhanda, M. and Pande, S. (2020), “Efficient toolpath planning for voxel-based cnc rough machining”, Comput. Aided Des. Appl., Vol. 18 No. 2, pp. 285–296, https://doi.org/10.14733/ cadaps.2021.285-296.

Maturana, D. and Scherer, S. (2015), “Voxnet: A 3d convolutional neural network for real-time object recognition”, in: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 922–928, https://doi.org/10.1109/IROS.2015.7353481.

Ranscombe, C., Hicks, B., Mullineux, G. and Singh, B. (2012), “Visually decomposing vehicle images: Exploring the influence of different aesthetic features on consumer perception of brand”, Design Studies, Vol. 33, pp. 319–341, https://doi.org/10.1016/j.destud.2011.06.006.

Real, R., Gopsill, J., Jones, D., Snider, C. and Hicks, B. (2021), “Distinguishing artefacts: evaluating the saturation point of convolutional neural networks”, Procedia CIRP, Vol. 100, pp. 385–390, https://doi.org/10.1016/j. procir.2021.05.089.

Su, H., Maji, S., Kalogerakis, E. and Learned-Miller, E. (2015), “Multi-view convolutional neural networks for 3d shape recognition”, in: Proceedings of the IEEE international conference on computer vision, pp. 945–953, https://doi.org/10.48550/arXiv.1505.00880.

Zaki, H.F., Shafait, F. and Mian, A. (2016), “Modeling 2d appearance evolution for 3d object categorization”, in: 2016 international conference on digital image computing: techniques and applications (DICTA), IEEE, pp. 1–8, https://doi.org/10.1109/DICTA.2016.7797065.

Zou, Z., Shi, Z., Guo, Y. and Ye, J. (2019), “Object detection in 20 years: A survey”, arXiv preprint arXiv:1905.05055, https://doi.org/10.48550/arXiv.1905.05055.

https://doi.org/10.1017/pds.2023.21 Published online by Cambridge University Press