Artificial Intelligence for the Built Environment

Sherif Tarabishy, Marcin Kosicki, and Martha Tsigkari

Abstract Artificial intelligence (AI) and machine learning (ML) have been repetitively mentioned in the headlines and associated with a multitude of disciplines and feats. So, what are in fact AI and ML and why should architects and engineers care? This chapter will address the above questions by explaining the meaning behind the buzzwords, clarifying the differences between AI and ML and discussing what applications they might have in the creative industries in general and in the construction industry in particular. Different learning paradigms and examples of what are the types of problems ML can solve in the construction industry are provided. Emphasis will be given on the quantity and quality of the data required to train a ML model, and how this is likely to be mined and curated within architecture and engineering firms in the next few years. In addition, two case studies—which focus on two types of ML models, namely surrogate and design-assist models—will be disambiguated, looking into why one might choose to utilize ML techniques to tackle a problem, the methodology to follow and how to critically evaluate the outcomes. One of the case studies deals with passively actuated laminates, conducted in collaboration with Autodesk, whilst the other with spatial and visual connectivity analysis of architectural floor plans. Finally, a discussion is drawn about present and future ML endeavours undertaken by the authors within their practise and how machine learning could be incorporated in the future of architecture and construction industry.

Keywords Machine learning · Artificial intelligence · Surrogate models · Design assist · Data mining · Data curation

S. Tarabishy (✉) · M. Kosicki · M. Tsigkari
Foster + Partners, Applied Research & Development, London, UK
e-mail: starabishy@FosterandPartners.com

M. Kosicki
e-mail: mkosicki@FosterandPartners.com

M. Tsigkari
e-mail: mtsigkar@FosterandPartners.com

M. Bolpagni et al. (eds.), *Industry 4.0 for the Built Environment*,
Structural Integrity 20, https://doi.org/10.1007/978-3-030-82430-3_5

1 Introduction

William Gibson, author of sci-fi novel Neuromancer, famously said: “The future is already here, it’s just not evenly distributed.” And he is right. Most people, for example, when faced with the question of what artificial intelligence is, think of the Matrix, or the Terminator’s Skynet: intelligent machines that could—in a distant future—take over the world! But artificial intelligence is much more real than a character in a sci-fi movie—and it’s not (at least currently) planning for world domination. On the contrary, along with machine learning (ML), AI has been repetitively mentioned in the headlines in the past decade, associated with a multitude of disciplines and feats. From stories of companies using artificial intelligence to detect breast cancer and outperforming experts, to start-ups using machine learning to create self-driving cars, the list of ways by which these technologies are used keeps growing. Their momentum is building up exponentially, and one thing is for sure: this time, they are here to stay.

Interestingly, not only are these technologies widely used in a plethora of industries, but their success is built on every single one of us: or rather on the data we produce. We all use (and effectively train), a ML algorithm every time we listen to music on Spotify, buy something online or even watch Netflix. Every single action in our life is underscored by the production of data that is being mined and used in a variety of ways: from identifying spam email to defining our banking credit profile.

Some may say that the above seem disconnected with the architecture, engineering, construction and operation industry (AECO). This could not be further from the truth.

The richness of data that is generated through all the stages of the life cycle of the built environment could revolutionize the industry, if leveraged appropriately using machine learning. This chapter will expand on how this may be possible by firstly explaining what ML is and how it works. It will subsequently frame it in the context of AECO, by providing examples of the data produced from conception to operation and how it can be used to train ML models. This will be showcased with theoretical and practical examples, investigating not only current applications, but also the potential of these technologies in the construction industry.

Using data to train machine learning models could serve AECO in a variety of ways, from helping designers and engineers deliver mundane tasks quickly, to performing like a design assistant and effectively enhancing or even showcasing creativity within the context of the built environment.

2 Machine Learning

2.1 Artificial Intelligence

Before expanding on machine learning, it is necessary that it should first be placed in the wider context of artificial intelligence. AI as a scientific field of study is quite expansive: its goal is to create machines that can simulate human intelligence and behaviour. To achieve that there are many subfields that need to be perfected together, each vast in and of itself. Some of these fields—besides ML—include but are not limited to natural language processing, robotics, computer vision and speech recognition. Figure 1 provides an inexhaustive list of research fields that are commonly attributed to the progression of artificial intelligence research. It is also worth noting that a lot of overlap exists between those research topics.

Not only is AI expansive, but it is also vague: its goal keeps getting redefined. Fifty years ago, an application running on a device the size of our palm that could play chess, and defeat its human (Chess Grandmaster) counterpart would have been described as AI. But nowadays, this is yet another app on a mobile device.

As technology and the understanding of how the human mind works progresses, AI's scope and what it should be capable of is reshaped and redefined constantly. In parallel, the optimism around AI and what it can accomplish waxes and wanes.

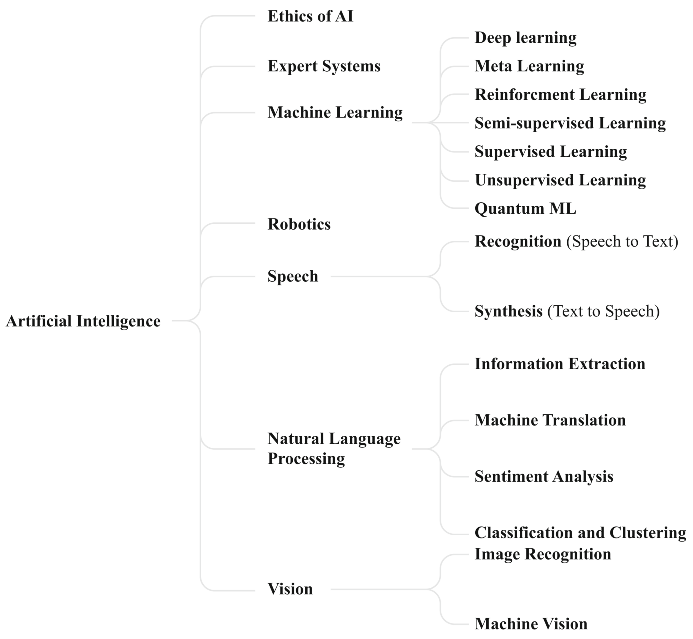

Fig. 1 An inexhaustive list of research fields that are commonly attributed to the progression of artificial intelligence research

Two more times in the past overhyped enthusiasm about AI was seen (in the 60s and 80s), both followed by reduced funding and disinterest in research when the predicted high expectations failed to materialize—periods known as "AI Winters".

Machine learning's scope is more targeted and considered a subset of AI as seen in Figs. 1 and 2. As summarized in the words of computer scientist Tom Mitchell: "Machine learning is the study of computer algorithms that allow computer programmes to automatically improve through experience" [1]. In ML, the aim is to develop a model that fits a specific type of problem [2]. To do this, an AI system needs to be able to extract patterns from raw data. As with any field of scientific study, there are different research approaches on how to accomplish this, materialized as algorithms and computational methods.

One such approach that has been gaining a lot of traction and media attention lately is the use of artificial neural networks (ANN). ANNs are loosely based on the neurons in the human brain and how they interact together. Collectively, they try to perform a bit like the human brain: taking data as an input, processing it through a network of neurons and finally outputting a response as seen in a simple network in Fig. 3. The neurons—structured as an array of consecutive *hidden layers*—decide whether to be activated or not based on a function that considers the weighted inputs of the previous layer. After that, their output propagates as an input to the next layer. This is done until we reach the final output of the network [3].

Historically, ANNs were limited in the number of neurons used within a limited number of layers. It was obvious though for some, since the 80s that the success of neural networks was intertwined with mainly two things, fast enough computers and big enough datasets [4]. With the satisfaction of those two conditions in recent years a new subfield of research rose: deep learning (DL). DL is a subset of ML research that is concerned with scaling up neural networks' algorithms. This is achieved by scaling up the size of the networks and the amount of neurons they include, by increasing the amount of data they could consume and by dealing with the by-product of this scale-up which is the amount of necessary computations.

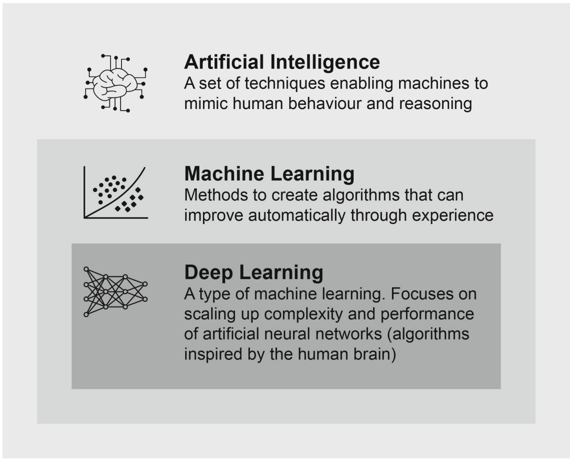
Artificial Intelligence
A set of techniques enabling machines to mimic human behaviour and reasoning
Machine Learning
Methods to create algorithms that can improve automatically through experience
Deep Learning
A type of machine learning. Focuses on scaling up complexity and performance of artificial neural networks (algorithms inspired by the human brain)

Fig. 2 A *diagram* showing the relationship between AI, ML and DL

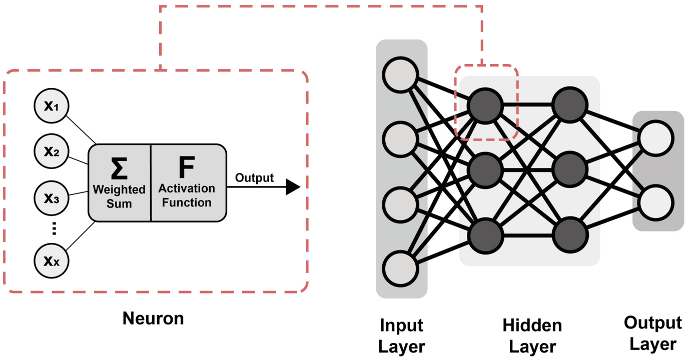

Fig. 3 A simple *fully connected* neural network, with a highlighted *neuron* structure, showing that each neuron's output is the weighted sum of all inputs, which then passes through an activation function that decides whether to intensify or abate the neuron's output

This chapter is not by any means an exhaustive introduction to ML, but rather an attempt to provide an overview of important terms, basic intuition of how neural networks work, the necessary logistics for using them and positioning those attempts within the wider scope of the AEC industry. Readers who find this interesting are encouraged to consider more comprehensive machine learning textbooks [5–8].

2.2 *Intuition*

So how does an artificial neural network manage to learn? It could be said that in very broad strokes an ANN develops similarly to a brain. Like the brain in a new-born child, the processing mechanisms are not useful at the beginning, and the response is far from being accurate. But through a process of feedback, a learning cycle is established.

To help create a mental model of this, you should imagine trying to balance a pen on your index finger. You might start by placing a random point of the pen on the side of your finger, getting a feel for how the weight of the pen is distributed as you are trying to balance it. Expectedly, at the beginning it will fall. You then start testing points to the left or to the right of the point you initially chose. Every time the pen falls, you keep building a better understanding of which point across the pen you should try and balance it from, until you reach your goal. You can memorize the optimal balancing position along the pen, but then most probably you will fail if you try again with a pen of a different size and weight balance. You might, weirdly, decide to take this task as a challenge and start trying hundreds, maybe thousands, of different pens! In ML terms, this phase is called the *training* *phase*. Once you can perfectly balance almost all of them, you are on your way to becoming an expert “pen balancer” (a coveted title!).

Congratulations: this expertise now allows you to be able to guess from a pen’s shape, profile and cross section, how the weight is distributed, and which point along the pen would make it balance perfectly on your finger! This phase is called *inference phase*, and the quality of your guesses will be highly dependent on the quality of your training. This tedious learning process of trial and error is exactly what an ANN tries to imitate. The nuances of this learning process are captured and modelled differently in different ML approaches (see Fig. 4).

With its capability to make accurate predictions, if trained on a good dataset, ML enables us to depart from traditional ways of solving tasks and detecting patterns which historically depended on explicit engineering and programming. For the sake of argument, let us assume simplistically that a robot is tasked to achieve the above

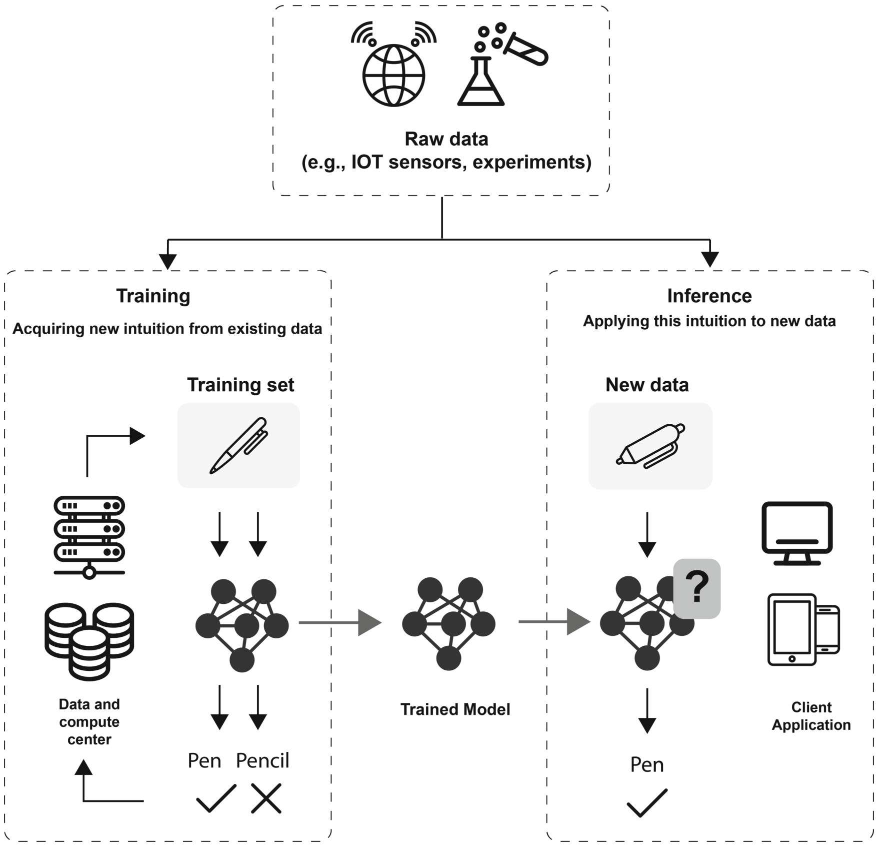

Fig. 4 During training, the model is continuously exposed to new data from the training dataset, and its knowledge is put to the test. Once we have a trained model, it can be used for inference/ prediction on new data. Only a subset of the data available is used for training

task. To programme this, first, one would need to collect data about different brands of pens, their length, cross section, weight distribution or any other features that might be deemed of importance to the task. Second, comes the task of examining optimal points along the pen where it could be balanced from. Then, one might start writing a very long procedure that gives explicit conditional instructions to the robot, manually encoding the knowledge about the pens that were surveyed. The programme can be quite long, but at the end, given a pen from the surveyed ones, our robot can potentially balance the pen on its finger. But what happens if there is the need to use a new pen or maybe a paint brush? Well, since an explicit definition of the desired output for every possible input was required, the programme would need to be rewritten or expanded, requiring further investment of time and labour. This is not always possible, and it certainly is not efficient. That is where ML shines: using ANNs, one could train the robot to perform the task on its own, without having to provide explicit and exhaustive instructions. This assumes though that one has the pens to begin with: that is, one has the *data*—loads and loads of it! The ANN will take as an input the pen, and output to the robot which point along the pen it should balance it from. Not only that but after a whilst it will even generalize, so if a decision was made that the robot needs to balance any object with properties like those of a pen, it should still be able to guess where the balancing point lies along this element.

2.3 Representation

But how can the ANN *take the pen as an input*? This question highlights an integral point in ML research, which is *representation*. A machine learning algorithm/model cannot really see, hear or sense the input directly. Some thought needs to be put into how to represent the input data in a way that the model can then consume. More importantly, one needs to think in advance which of the input qualities are key for the model's understanding of it (in order to complete its task), and consecutively shape the input dataset accordingly.

Figure 5 shows a classification problem, which in this case is whether an input presented to the system is a villa or not. Classification—one of the tasks ML is very good at—is used to figure out which category an input most likely belongs to, given a list of categories presented to the model (in this case "villa" or "not villa"). Using traditional approaches, first, one would need to figure out what are the key aspects that can portray the input (what makes a villa?). Second, one would need to provide an exhaustive list of instructions about how to detect whether those key aspects exist in an input or not. Given the above, a programme would be able to classify whether a given input is a villa.

ML algorithms can change this process, where instead of an exhaustive set of instructions, one needs to only provide an algorithm that works for this particular task—e.g. a logistic regression model or naïve Bayes model or a similar ML classification algorithm [9].

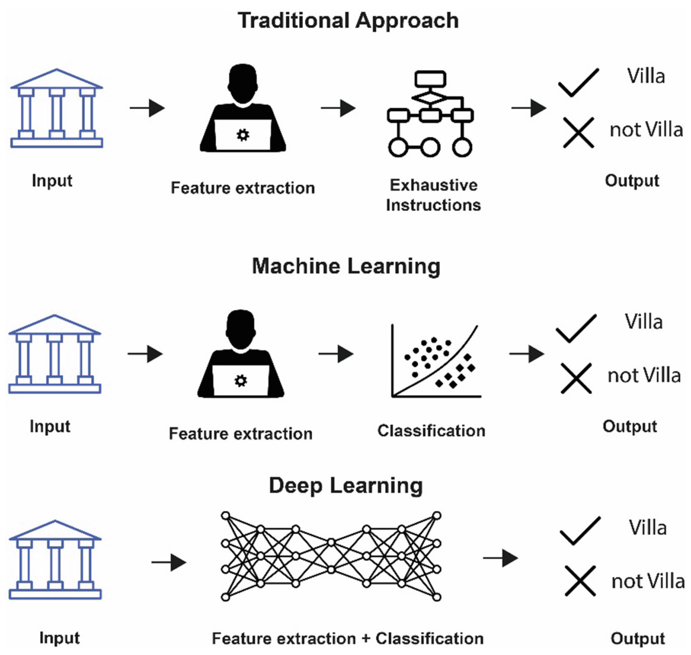

Fig. 5 Diagrams of a classification problem solved by a traditional approach, machine learning and deep learning, respectively

Deep learning or deep neural networks have taken this process even further, where the model is not only capable of learning the representation (feature extraction) best suited to portray the inputs for the given task, but given a new input, it can figure out which category it belongs to.

Going back to Fig. 3, we can assume that our dataset is comprised of images. In this case (Fig. 6), each pixel in the image would be one of our inputs. If a low-resolution image is used, e.g. 35 × 35 pixels, this means that the total number of inputs would be 1,225 (35 × 35). But an image usually has 3 channels (red, green and blue), each of which need to be an independent input. This would lead to an input size of 3,675 (1,225 × 3). The neural network referenced in Fig. 3 is called a *fully connected network*, since each neuron in a layer is connected to all neurons from the previous layer. If our network has only a single hidden layer with 10 neurons that would result in a total number of connections of 36,750. One can see how this number will drastically increase and become unmanageable if higher resolution images are used or if more hidden layers and more neurons per layer are added (deep NN). Another problem is that the now *flattened* input (which has been transformed from a 250 × 250 × 3 pixel grid to a 3,675-long list of input values) has lost all spatial information in regards to each pixel's spatial relationship to its neighbours. This was one of the many reasons why a different architecture of neural networks was needed. Convolutional NNs started out as an adaptation of ANNs to

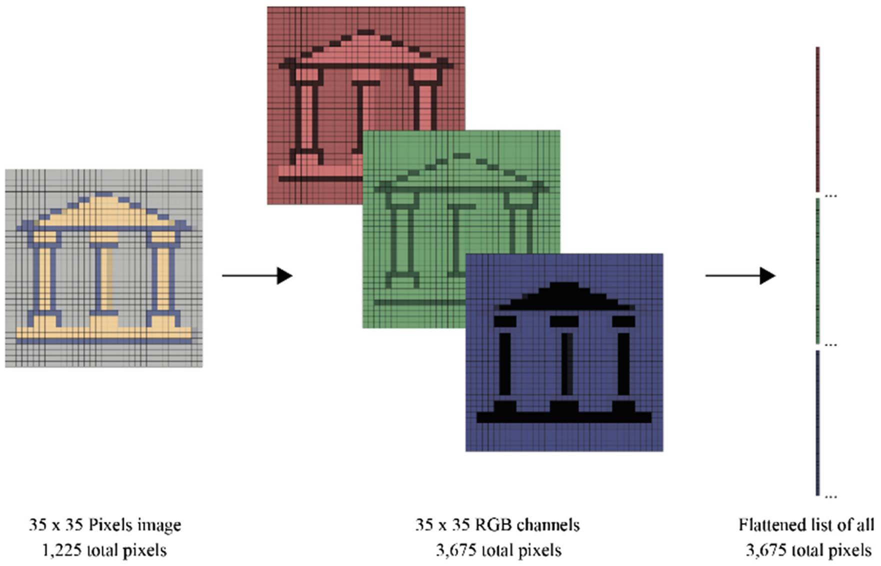

Fig. 6 A diagram showing the amount of input parameters a fully connected network would require for an image of size 35 × 35 pixels

work with image-based tasks, but later their use expanded to other domains. It was inspired by classical image processing techniques; it tries to limit the number of neurons and connections by sharing them and utilizes other optimizations to deal with image data—the mechanics of which are outside the scope of this chapter.

For each domain and task, variations of the classical ANN exist. Some more suited for a type of data input than others. There are particular ANN models that specialize in image recognition [10, 11], text generation [12], audio synthesis [13], graph structures [14] and 3-dimensional representations like voxel data [15], point-cloud data [16], mesh data [17], SDF data [18] and many more.

2.4 Learning Paradigms

In an interesting parallel to real-life, there are different “styles” (or in this case *types*) of learning used within the context of ML. These are usually split based on two main differentiators: (1) whether a teacher or some form of feedback exists in the process or not and (2) whether the learning entity is in active learning state (exploring the environment) or passive learning state (being presented with data, without interacting with the environment).

Table 1 shows the different types of learning based on the above [19]. These are not the only styles of ML but the below split is the one most commonly observed.

Table 1 Learning paradigms

	With feedback/teacher	Without feedback/teacher
Active	Reinforcement learning/active learning	Intrinsic motivation/exploration
Passive	Supervised learning	Unsupervised learning

In the next few paragraphs, an effort will be made to explain these four basic learning paradigms with simple examples.

The following learning styles had—through the years—various computational models implementing them. With the rise of DL though, some of those tasks vastly outer-performed classical ML algorithms. That by no means makes DL the best solution for everything, as depending on the task and amount of data used, its use may be considered an overkill.

Supervised learning. This in ML would be the equivalent of a teacher setting up assignments for a class, for each one of which he/she has the perfect answer. Thus, whenever a student submits their assignment, the teacher can provide detailed feedback, of whether the answers are right or wrong, and if the latter, by how much, so the student can adjust their knowledge of the subject and become better at it. In this supervised learning approach, the model (student) is presented with a dataset of inputs and their respective outputs (ground truth). This means that during training, for each input the model can also be provided with its error value, calculated as the difference between its output (prediction) and the ground truth. This process continues, where the model is "fitted" to the training data, ensuring it can predict the answers for almost all questions correctly.

When it comes to supervised learning, data curation and testing are both important parts of the process. Going back to the teacher-student analogy, let us assume that the teacher is trying to prepare handouts to help the students achieve top marks. To do so, the teacher would have to prepare extensive training material of data carefully curated (curriculum) to impart the knowledge the students would require in order to get high marks in their assignments. In ML, this would be the equivalent of the data required to train the system. This data cannot be presented at random, but rather must be carefully selected, cleaned and organized before it is input to the ML model. This data curation is an essential step in feedback-based ML —and is usually the hardest.

As mentioned above, testing is also important in ML. In the case of the teaching example above, it would be common for the teacher to set small quizzes to test the student's understanding of the subject at hand. In ML, this happens through setting part of the dataset aside, in order to conduct a test after the training process ends and check the model's predictive ability on data it did not encounter during training. This data subset is called the *testing set* as opposed to the *training set*, and it is an incredibly important part of supervised learning. It is critical to use diversified data sets to validate and test an ML model to avoid over fitting [20] leading to the model not generalizing well. Generalization refers to the ability of the model to correctly predict answers to questions it did not see before, questions from outside its training set, which is the ultimate goal.

A good example of supervised learning in the AECO industry would be the T2D2: Damage Detector created by Core TT [21]. The application in part uses a model trained through supervised learning to detect and classify damage to structures and materials given an image. This approach saves a lot of time and manual labour. The user can provide hundreds of images for the state of the building taken using reality capture solutions. The service then uses those images to construct a 3d model of the building and highlight where damage is seen and what type of damage it is.

Reinforcement learning. This is effectively a type of feedback-based ML where a fixed dataset to train against is not available. It would resemble a player finding an obscure arcade game with no instructions. In this case, if all the player had to go by was the controls and the scoreboard, they would use the former to explore the gaming environment and the latter as a metric of their success. The higher the score, the closer they would be at figuring out the rules of the game. As opposed to supervised learning, where a model answer (desired output) to a given input is available, in reinforcement learning there are no model answers; there exists only the ability of assigning a measure to how good the resulting output is, or in other words one can "score the behaviour of the model according to some performance criterion" [22]. In addition, since the learner here is active within the environment, we refer to it as the *agent*. Just like with the player, there is no known answer—no map to guide the agent through the rules: there is also no way of knowing which exact actions during the game are beneficial or not, since the score is not necessarily updated with every one of them. This sparse reward signal might affect the agent's learning progress.

What this means is that in reinforcement learning, the agent—just like the player in the above example—will try to come up with its own playing technique: a system to help provides guidance on which action will yield maximum rewards given the current state of the game/environment. Leveraging memory, eventually, the search efforts of the agent decrease with the increase in its experience.

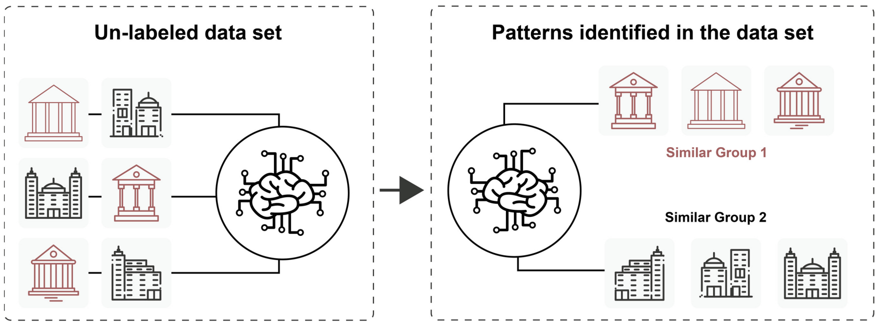

Fig. 7 A diagram showing clustering which is an unsupervised learning task

A great real-life example of this type of learning is DeepMind's Agent57, a model trained to play 57 Atari games that managed to obtain scores higher than the human baseline scores.

Unsupervised learning. This type of learning comes into play when there are not available desired behaviours (outputs) for a given input, nor is there a way to judge if a behaviour is good or bad based on a performance criterion. This technique is usually concerned with uncovering structure and finding patterns hidden in unlabelled data (Fig. 7). In the teacher analogy, it would be as if the teacher had some questions that they did not know how to answer, about some random uncurated content and they just dumped all of this on the students. This sounds like a bad teacher indeed, which is why unsupervised learning is associated not with a teacher, but the lack of one thereof.

Some of the tasks unsupervised learning is good at are clustering and dimensionality reduction. Clustering has interesting repercussions, particularly when compared to classification (the typical supervised learning task explained in Sect. 2.3, some examples of which are the k-nearest neighbour (KNN) model, logistic regression, naïve Bayes classifiers and random forests). In both cases, the goal is to split huge amounts of data into groups, but whilst the later tries to sort it on a predefined set of clusters, the former is making no assumptions of how the split should happen. On the other hand, dimensionality reduction tries to provide a more readable organization for data with lots of attributes. An example of that would be sorting pixels with different RGB values (three dimensions) in a 2D plane (two dimensions). Or taking multidimensional objects and representing in a lower dimension (2D grid) based on the relationships between their inputs (see Fig. 8).

A good case study of unsupervised learning implementation can be found on how Netflix operates [23]. In very simple terms, prior to 2016, Netflix was building its viewer recommendations based on gender, age, location and other demographics —performing, effectively, a classification problem. But these predetermined categories were proven to not be really indicative of which type of content viewers enjoyed. Since then, Netflix started collecting all sorts of behavioural information: from the time a viewer watched a movie to the amount of time one may spend browsing content before choosing something to watch. Based on all of that extra

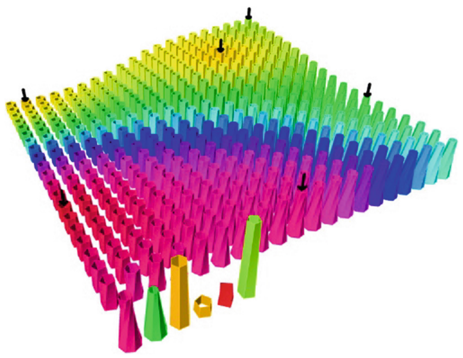
Fig. 8 Unsupervised learning: dimensionality reduction of a multi-variable massing problem to a 2D representation using a self-organized map (SOM)

data, users are now clustered into “taste communities”, which are categories developed based on data, and are currently used to tailor recommendations with much greater success than before [24].

Intrinsic motivation and exploration learning. This final type is an interesting ML paradigm investigating active ML models that do not require feedback. This model is trying to address the main problem of supervised learning, which is that learning through extrinsic rewards alone is not how humans evolve their knowledge. And that’s because despite the fact that humans do learn based on “rewards” as they are exposed to an array of experiences through their life, they still have the capacity to navigate through first time experiences without any prior knowledge or direction whatsoever. Intrinsic motivation and exploration are based on this premise, whose focus is to try to generalize and develop models that work well at a wide variety of tasks and environments—tasks the models haven’t been exposed to before, and yet could still manage to navigate autonomously, making this field of research a hot topic of investigation [25].

3 Architecture, Engineering, Construction and Operation

3.1 Data

If machine learning was an engine (a high consumption one—think Lamborghini), then data would be its fuel: it would need loads of it, and of the highest-octane variety for that matter! When it comes to data for training ML systems, good things cannot come in small packages—quantity and quality are equally important.

As a by-product of everything humans do, data is embedded in every one of their actions. The previous sub-section explored how some of that data can be used to train an ML model in different ways. But what about the AECO industry? What are the types of data that can be extracted from the built environment? It is safe to say that the multidisciplinary nature of AECO can promise a very rich data landscape, comprising everything from variety of layouts on floor plates to operational information available during the life cycle of the building.

Some of the aspects of AECO data are covered in chapter “Big Data and Cloud Computing for the Built Environment”, where data gathered in the built environment is effectively broken down in 3 temporal subcategories based on the stages in the life of a building: design, construction and operation. During these stages, anything that describes a building, how it is made and how it operates, can be collected as data. But what is more interesting, is not just the type of datasets that can be extracted, but rather how they are related to each other. It is this understanding that makes its leverage more powerful. For example, a dataset of generic massings of buildings relative to their plot outline could be collected to train a system. Assuming that after training the system will, theoretically, be capable of producing a 3D model of a building, given a plot as an input. How does the user know though that this building is relevant to the plot? The dataset would require further refinement: further data should be collected on building regulations, built context, environmental analyses, amongst others—data that not only is related to that particular massing, but also to its performance and context. The models used as data would not only need to be comprised of optimal individuals (massings that do perform well in terms of daylight for example or comply to the building regulations given their geolocation) but also the system would need to be trained in association to the multiple datasets describing each massing.

As mentioned before, everything humans produce can be leveraged as data. What is challenging—and particularly in AECO—is how the right datasets are identified and brought together, so the system (ML model) can make interesting correlations and produce performance driven results—whether that performance is judged by hard criteria (sustainability), or soft criteria (aesthetics).

Machine learning data is the crux of the matter: the success of the system's training is only as good as the data it is provided with. In essence, the data used to feed a ML model can have a significant impact on its performance [26]. Most failures in ML trained systems will be a result of poor or bad datasets (the process "affectionately" known in computer science as GIGO: garbage in, garbage out). There are of course other factors that will affect how effective the system is, like the choice of the ML model used (e.g. decision trees are great for interpretability, but as they are prone to over fitting—modelling a lot of the noise in the training data—and computationally intensive, they might not be the best choice for very rich and big datasets), the choice of the cost function (which describes how the error rate between the predicted and expected values is calculated) or how the model deals with its bias-variance trade-off. These are only a few of the components of a ML recipe, but the most significant ingredient is—as mentioned above—the quality and quantity of the input data used. Small datasets can be prone to bias, outliers in datasets can result in over fitting, poorly labelled datasets can lead to misleading outputs, skewed distribution of data will produce errors—the list is endless. In any ML endeavour, the hardest task is having good datasets to train a system on.

3.2 Tasks

Data may be abundantly available in AECO, but that does not mean that it is as easily accessible or retrievable. And that is because although a huge amount of data is produced during the design, construction and operation of the built environment, this is usually scattered and unstructured. The industry's current pipelines do not offer a straightforward way for the data that is produced to be collected, processed and classified in a meaningful and useful way. This is due to an array of factors, having to do, not only with the vast variety of formats used (sketches, drawings, models, reports to name just a few), but also the nature of the data itself. Whilst it's straightforward to collect and process set of images that represent animals for example, it is much more difficult to extract useful and comparable information from a 3D model (not to mention the difficulties arising from what software this model was created in). This information can be related to anything, from programme and overall areas, to layouts, materials, simulations related to the model, structural or environmental performance, occupancy, energy consumption, embedded and embodied CO_2 to name a few.

So, the industry is faced with an interesting challenge: how can data be collected, organized and processed across disciplines during the life of a project, from design to operation, in a meaningful manner? And ultimately, how can this data be leveraged for future projects—how can it be gathered and used in a structured way further down the line? Obviously, the latter is not anything new in the profession: every experienced architect, engineer and contractor are doing this implicitly, building their knowledge on top of the experience acquired on every project they worked on. But this type of knowledge that lives in the head of every experienced professional is, unfortunately, not easily transferable.

Remarkably, where a far better collection and use of data is observed in the industry, is during the operational stage of the building. Interestingly, this task is usually the job of facility managers or other consultants rather than architects. Furthermore, the proprietary nature of the data makes it inaccessible to the project's architects.

The integration of sensors and the internet of things (IoT) in the built environment has given rise to more and more architects and clients adhering to the idea of smart buildings. Data collected during the life of a building, which may not only be related to the building itself (e.g. thermal comfort or energy consumption) but also to its occupants (e.g. meeting room occupancy), allows us to monitor and measure its performance not only quantitatively, but more importantly qualitatively. This data is then usually fed to ML driven building management systems (BMS) that can regulate how the building performs, make suggestions or even identify issues. It may advise on things such as optimal use of meeting rooms, or optimal number of different functional spaces for a building of such capacity. All that information, when captured, becomes then a driver not only for the current building, but also can be leveraged in every building project after that.

It is interesting here to point out that this is exactly what the AECO pipeline is lacking during design: a way of not only capturing the data, but a method of being able to reuse it in the future projects. Most of that knowledge is transferred as experience through the architect's and engineer's mind, but how can it be stored as a hard metric that could be used for every project that will follow? This is the big question that the industry is currently called to address—and the answer is not straightforward.

3.3 *First to Market*

There are already various AEC start-ups that are touting machine learning as part of their offering. Some focused on site and construction like Insite [27] and Built

Robotics [28] who are looking into upgrading construction site's heavy machinery with AI-based capabilities, increasing safety, guiding operators to use most efficient trajectories for a given task or even allowing the machines to operate in a fully autonomous mode.

Some applications focus on making long running analysis faster by providing close enough predictions like the Austrian Institute of Technology's wind flow simulation integrated in Giraffe [29] or bringing forward analysis that would usually be conducted in later stages of a project's development, like Core TT's Asterisk [30] for structural analysis. Others are trying to leverage the research being done with natural language processing (NLP), like UpCodes [31] who is creating a tool for compliance of 3D models with building codes, by automating design reviews and document control. Some platforms, like Archistar [32] and Spacemaker [33], are even looking into providing designers and stakeholders design aid during the early stage of the design process.

One commonality between most of those offerings is that the services are in the form of cloud-based web platforms, forcing users to upload their data to where the models are hosted if they are to use the trained models' functionality. This might prove problematic with some users, as maintaining the privacy, security and integrity of the data becomes a general concern. On one hand, the companies offering those services wouldn't want to divulge their intellectual property by sending the model that they developed and trained to the end-user [34] opening up the possibility of reverse engineering attacks [35], nor would they want to lose the opportunity of a usage-based pricing model. On the other hand, the users prefer keeping their data private and secure in their servers, particularly when client confidentiality issues come into play. In quest for a solution to this issue, a new way of dealing with said data—called *federated learning*—is getting a lot of focus [36]. One of the main objectives behind federated learning is to train a central model with data distributed over several machines, whilst maintaining the privacy and integrity of each users' data. Several organizations are starting to support this approach, most notably OpenMined [37] which is an open source community creating an array of tools and techniques to aid researchers and practitioners in adapting this learning type in production.

3.4 Subjectivity, Creativity, Bias and Interpretability

Imagine that a user wants to build an application to recognize—let's say—a Palladian villa. To do so, one needs to select the ML model to use (in this case a convolutional neural network (CNN) would be a good choice, as they represent a type of deep neural networks commonly used to analyse images, see Fig. 9) and feed the system thousands of images of Palladian villas, training it not to regurgitate them, but rather to identify from them what are the defining characteristics of Palladian villas and thus be able to recognize them (or not) when a new image is presented to it. What is important in this process is that the user does not explicitly

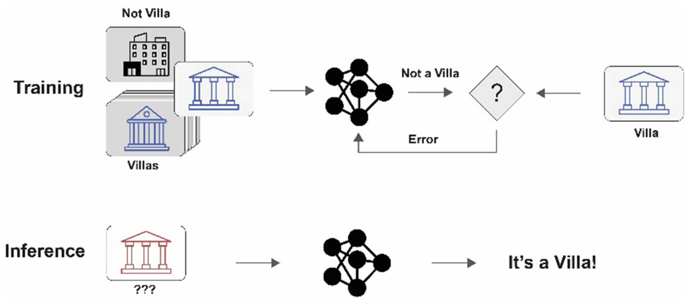

Fig. 9 Training and inference of an ML-based system for recognizing Palladian Villas

specify the characteristics of Palladian architecture to guide the system: rather the system manages to extract these traits autonomously.

The above example may seem redundant if one is familiar with Palladian Architecture. Through the years, this typology has been studied by different people, which resulted in a detailed description of the rules and concepts that constitute a building of that typology. But what makes the ML method, interesting, is that it doesn't require the huge amount of people, time and effort spent in encoding all this knowledge into an exhaustive programme that once presented with a building, it would be able to tell whether it follows the Palladian rules or not. If nothing else, this process manages to save the user a huge amount of time and money. But this is not what makes ML a great tool. Things become fascinating when one starts posing questions for which they do not have an answer. What if, for example, what needs to be identified or predicted is not governed by a well-defined ruleset? What if one wanted to incorporate and capture more subjective, elusive or unpredictable qualities? In face of those questions, traditional approaches fail. A user may know what the defining characteristics of a Palladian villa may be, but can they pinpoint all the characteristics of a successful public plaza? They may intuitively (or based on experience) understand some of them, but if they train a system based on a set of thousands of—*what they think are*—successful public spaces that system could then be tasked with generating other spaces that are similar in traits. In many cases, this training process comes up with incredible results: characteristics and correlations that would be too obscure or complicated for the user to pick, formulate and encode independently.

A common pitfall, when dealing with ANNs, is subjectivity. Hopefully, *what they think are successful* in the previous paragraph gave you pause. There are a lot of prominent examples that call attention to the issue of bias in tools and apps that are being used daily by the public. One such example is Twitter's algorithm that it uses to crop and centre the content of posted images: users were dismayed to find it favouring white faces over darker ones [38]. Similarly, Amazon has stopped the development of an ML driven tool used for recruitment and filtering job applications after they discovered it was gender-biased towards male candidates [39]. The above showcases how easy it is for the datasets used to train these systems to introduce error and biases, which become a significant issue when this technology is spreading to areas like medicine, law and transportation to name a few. One approach suggested by researchers was to require companies and researchers to produce public and transparent data statements that can be audited and challenged [40], which may be a way to alleviate the issues.

Another catch: the fact that ANNs utilizes correlation between variables, means that conditions existing in our dataset during training should remain consistent when the model is used to make predictions based on new input data. Imagine, for example, the previously described model that generates building massing, after it was trained on building regulations, built context, environmental analyses, etc. What would happen if the regulations changed after the model was trained? The correlation between the inputs and outputs would be different. In this case, it cannot be assumed that the previously trained model would work as expected, as the model has no capacity of understanding causality. In fact, causality and casual predictions are a field that researchers are looking into as one of the ways of making better and more versatile models.

One more shortcoming of ANNs is interpretability: the ML trained model may be able to predict the design of a successful public plaza, but the reasons why this is so cannot be easily inferred. Neural networks are in many ways a black box, with justifications of its internal workings being totally obscured from the user. Unlike other ML models, like linear regression for example, where the evaluated coefficients of a polynomial can define the statistical significance of each of the variables, in ANNs the process of how a prediction is made is hidden. The input is connected to the output by one or many hidden layers of artificial neurons that process the weighted input signals through an activation function. But this process does not specify what is the correlation of the various inputs to the output. It is, therefore, very hard to evaluate the reasoning behind the different outcomes, making interpretability yet another field of research that is getting a lot of focus [41].

Nonetheless, machine learning could be a very interesting tool in the hands of the AECO industry. The built environment is multidisciplinary endeavour and traversed by a plethora of multi-objective criteria. The datasets (whether sketches, models, technical drawings, analysis data, BIM models, fabrication data, post-occupancy data, etc.) are rich and complicated. They are datasets that can provide invaluable information and help train systems for a variety of undertakings: automating mundane tasks (e.g. furniture layouts on a given floorplate), extracting relationships in design problems, providing design assistance or real-time analytics and insights about new projects built all over the world. All these capabilities would enhance the designers' knowledge and sensitivities, make them privy to correlations that are not obvious and ultimately free them from routine tasks, thus allowing them more time to be creative and helping them make more informed decisions.

4 Case Studies

With Foster + Partners always striving to be in the forefront of innovation, the Applied Research + Development group (ARD) has been investigating the potential of machine learning and how it could be incorporated throughout the practise's design pipelines. Thus, two ways by which ML could be directly useful in our current processes has been identified: surrogate models and design assistance models. Surrogate models, also known as approximation models, are used as a replacement for analytical engineering simulations, when the simulation might be computationally and temporally expensive to calculate. Real world design problems require simulations which can take hours, or even days to complete. This problem leads to tasks such as design optimization or design space exploration becoming nearly impossible since they require thousands to millions of simulations in order to converge. With the objective of providing designers with the output of those simulations near real-time, ARD started looking into creating approximation models, where a computationally cheaper predictive model is constructed based on several intelligently chosen simulation results. An example of this approach, where a model can predict results of a spatial connectivity simulation, is explained in more detail later in this chapter.

On the other hand, design assistance models fall more in the category of automating tedious processes, where one does not necessarily need to have an analytical answer. This could be, for instance, the spatial layout of furniture within a space or providing designers with document control assistance in real-time, whilst they are working.

In both cases above (surrogate models or design assistance models) data is a necessity to train the system. Generally, there are two types of data one can work with: *original dat*a or (plausible) *synthesized data*. Original data can be accessed through the office's huge database and is comprised of all the drawings, models, sketches, details, etc. produced over the 53 years of the practise's existence. Synthesized data is data which can be automatically generated with the help of a generative design system. Each has its own challenges. For original data—and depending on the task at hand—it can be quite challenging to sift through years of digitally archived work in different file formats, produced by thousands of employees for thousands of different projects and be able to find the data that would help us learn something about our task. On the other hand, synthesized data may not always be an option either, as creating generative models with the richness of information required may prove an almost impossible task.

4.1 Design-Assist Model

The first project presented here, conducted in collaboration with Autodesk in 2017, was inspired by the recent advances in material science—and more precisely the developments around smart, passively actuated materials. These materials can change their shape and deform under certain conditions without any help from external mechanical actuation. They react like living organisms by adapting to changes in their physical environment, e.g. thermal fluctuations, changes in light conditions or even humidity. These materials could have great potential in architecture. Imagine an adaptive façade that does not have any mechanical shading devices, but rather self-deforms based on external conditions—like a living skin—to accommodate requirements of shading, overheating or privacy. This could be possible by taking advantage of the material's layering: by mixing patterns of thermo-active materials around passive laminates, a difference in expansion and contraction rates occurs. That difference, if curated, can lead to designed deformations, which we, as designers, could control.

With that in mind and working very closely with Panagiotis Michalatos and Amira Abdel-Rahman at Autodesk, our Applied Research + Development group started investigating how laminates made from layers of smart materials behaved. As architects, seeking to use these materials for adaptive façades in the future, what was of interest was their morphological deformation: a controlled transition from some initial state to an end state and back. Since these deformations are both plastic and large (compared to relative sizes of the analysed objects), there exists a nonlinear relation between the laminates' internal forces and their displacements. This is opposed to a more common linear analysis, where force–displacement relationships remain constant. As a result, such problems require a more sophisticated and time-consuming simulation strategy for nonlinear analysis.

In this research, every laminate had an initial, non-deformed state (e.g. fully open, adaptive shading component) and a target deformation (e.g. closed façade). In a linear problem, the effect of tweaking input parameters on the output is predictable. But in this case, faced with a nonlinear problem, it is very difficult to predict how different parameters—e.g. the distribution of the thermo-active material over the laminate layers at the initial state—would affect the resulting deformation under a given temperature. One way this problem could be approached would be to guess an initial layering for the thermo-active material, run a nonlinear finite element analysis (FEA) to accurately simulate the laminate's structural deformation, then slightly change the laminate's layering and keep repeating the whole process hoping that the changes would yield results closer to the target deformation. This process is similar in nature to trying to balance the pen on your index finger, as explained before! In this case though, the process would have been extremely time consuming, given the amount of time it takes to simulate the laminate's structural deformation and could only yield grossly approximated results.

Therefore, as showed in Fig. 10, it was decided to investigate whether an inverse design problem could be solved. In this scenario, the final structural deformation would be the input, rather than the output, to a process that could then provide the initial layering which would cause such deformation. Since obtaining an analytical solution for this problem was not possible, it was decided to use machine learning to build a predictive model. As mentioned before, every predictive model requires high-quality data to learn from. Not having that original data right off the bat, it was

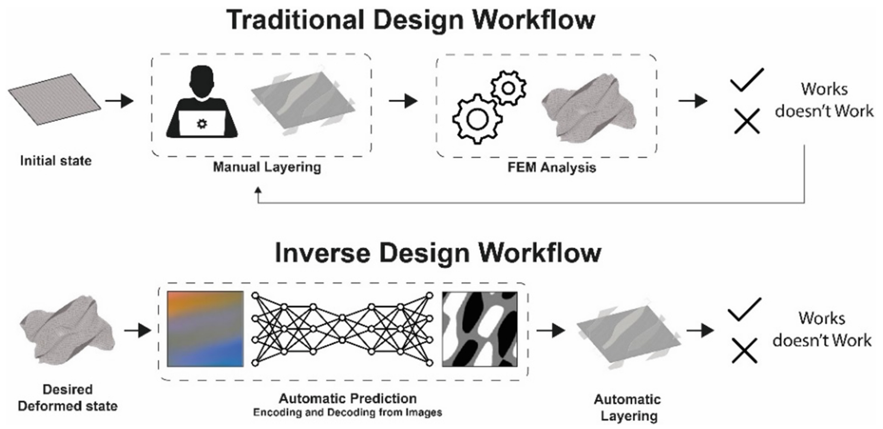

Fig. 10 Comparison between traditional and inverse design workflows

decided to go down the synthesized data route and create a generative dataset. To that end, a parametric model was developed based on a simple design intent which could generate variations of wavy layering representing laminates' initial states. Subsequently, hundreds of thousands of FEA simulations were run for a plethora of laminates sampled from the parametric model. The process of simulating the deformations of the newly generated samples was run in parallel and distributed (using an in-house custom-written software called Hydra) on an on-premises compute cluster. Hydra made possible to analyse the synthetic dataset tens of times faster than using any out of the box commercial modelling and analysis software. Later, this dataset (the deformations derived from the initial states) was fed into a system of two ANNs competing against each other.

To explain this, one could revisit the example of the expert "pen balancer", but with a twist! You are approached by a person who aspires to be your mentee. Interestingly though, the mentee's aspiration is not to be a pen balancer himself, but rather wants to design weird new pens, that can balance way off centre. They want to be able to pick a point along the length of the pen and design a shape that would make it balance on that point. The mentee knows that an expert "pen balancer" would be the perfect mentor as someone who has seen thousands of pens and knows where each of them should be balanced from. Being the good mentor that you are, you explain to your mentee the structure of the learning sessions: you ask them to pick a point along the length of the pen where they want it to balance at and draw a shape around that. As a mentor, using your expansive knowledge, you will provide feedback regarding two aspects. The first is whether the shapes they are drawing look like pens or not, and the second, whether that shape would balance around the point they picked or not. This loop of the mentee picking a point and drawing a shape based on that choice, and you providing feedback keeps going for a long whilst ("yes, this looks like a pen, but it would balance around this point, not that!" or "no, this looks nothing like a pen, but if you adjust this part maybe it looks a bit closer to a pen. And by the way, this is the point it will most probably balance on").

As the process progresses, the mentee gets better at creating pens that can balance wherever they may wish, but also the mentor gets better at understanding what the essence of a pen is and the physics behind how it balances, by virtue of researching more in order to be able to help the mentee. This structure of ANNs was proposed in 2014 [42] and is called generative adversarial networks (GAN), some variations of this structure are regarded as part of semi-supervised or unsupervised learning techniques, whilst other variations like the one used in this case study rely on framing the task as a supervised learning one. At some point, this adversarial system reaches an equilibrium where both networks cannot improve anymore, and the outputs generated by the mentee are hyper-realistic based on the mentor's knowledge. In GANs literature, there is often a different analogy used to describe this, and marking one of the models as an art expert and the other model as a forger. But that mental model obscures a very important part of the training process of GANs: the part where the art expert is in fact helping the forger get better!

In this collaborative research with Autodesk, this type of GAN was used to train on pairs of deformed laminates as inputs (the point at which the pen is required to balance) and cut-out patterns of initial laminates' layers which caused a given deformation (our pens in the analogy) as outputs (see Fig. 11). In literature, the mentor is called the *discriminator* and the mentee the *generator*. The discriminator has expansive knowledge, obtained by seeing all the layering designs generated and their respective deformation. The generator is given an image of the deformed laminates and then asked to start figuring out what the initial laminate layering should be. Again, the generator is not randomly giving answers, it is being guided by the discriminator.

The generator model was able to create acceptable layering for a given deformation within milliseconds after its training (see Fig. 12). This allowed our combined Foster + Partners and Autodesk team to prototype a simple yet novel

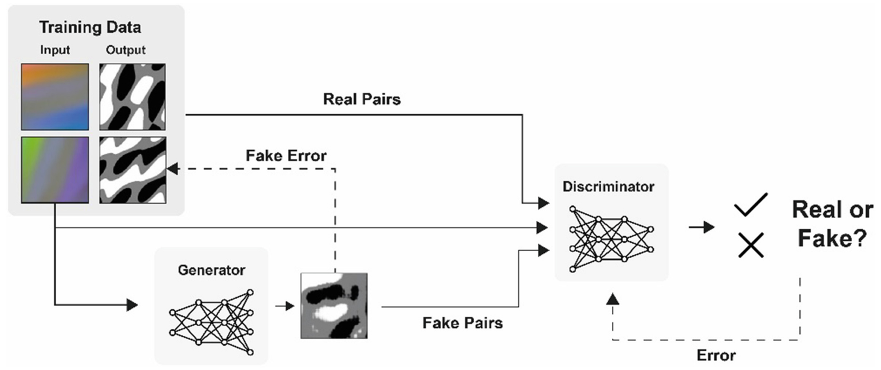

Fig. 11 Diagram showing the training methodology for the GAN-based system used for the passive material research project

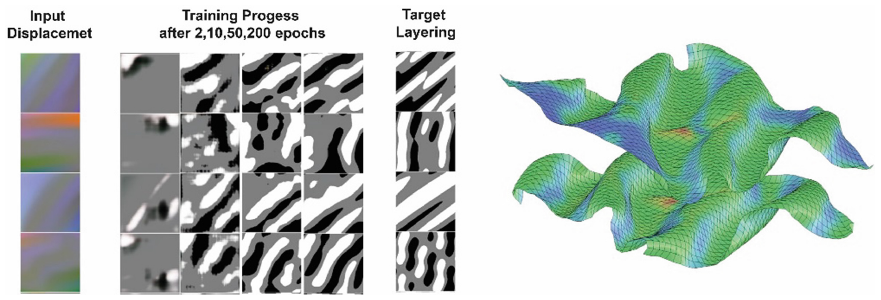

Fig. 12 On the left-hand side, progress of the GAN-based system learning to predict the layering based on a given input displacement. On the right, the difference between the ground truth deformation and the model's predicted deformation

application where a designer could interactively design a laminate in a deformed state and be presented with the cut-out patterns almost instantaneously. This workflow not only completely challenged the way such laminates are being currently designed, but it also suggested a methodology of significantly reducing the prototyping phase. The results (see Fig. 10) proved the effectiveness of the proposed workflow, especially at an early design stage [43].

4.2 Surrogate Model

From the design of the material, let's now jump to that of a space. And rather, how that space could be better designed based on performance criteria. Spatial and visual connectivity (or visual graph analysis—VGA) are two of the various metrics used to evaluate the performance of a floor plan. They are excellent metrics to help designers understand how well a floorplate works in terms of walkability, creation of public versus private spaces, instigation of serendipitous knowledge cross-fertilization and visual navigation.

The problem is that the two analyses can take hours to compute for large scale floor plans—particularly on high resolution. The goal was to significantly cut down this time to a near real-time experience, which would make the analyses accessible and intuitive for designers to use during their design process. Previously, state-of-the-art algorithms and parallelization techniques were used to get the analysis down from hours to minutes. Nonetheless, that was not anywhere near real-time performance. So, the possibility of training a surrogate model to approximate the analysis output was investigated.

Like the previous case study, a parametric model capable of generating basic floorplan variations was developed, which incorporated open plan and compartmentalized office spaces, complete with walls, doors and furniture (see Fig. 13). This model was then used for generating a dataset of thousands of synthetic floor

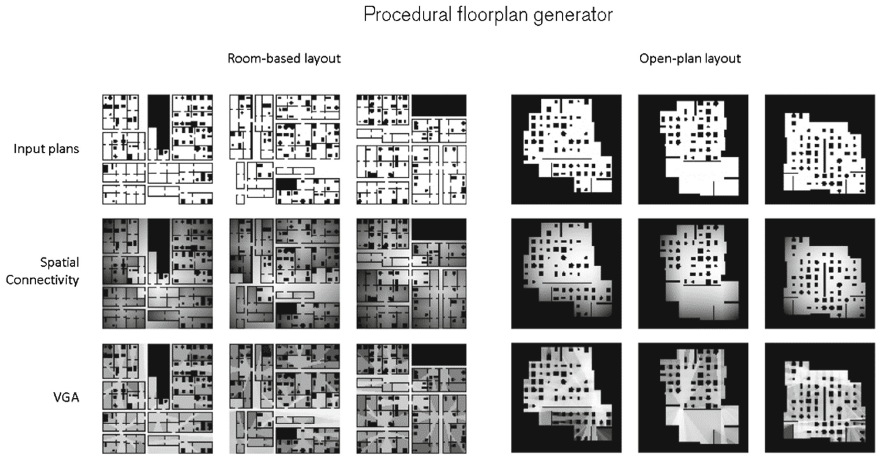

Fig. 13 Generated floorplates and their respective analyses for the connectivity study

plan images. Using the analytical accurate engines that were previously developed in-house for both analyses, the analyses were run on those synthesized images. This step would have taken days, but again thanks to ARD's bespoke distributed computing software called Hydra—mentioned previously—it took hours. The result was a curated set of floor plan images that would be used as an input for the ML model and the same images with the analyses result overlaid on top, which will be the output the model is asked to generate.

Now, the ML model was supposed to answer one tough question: given those thousands of images, what is the function or process needed, in order to map correctly between each input and output image in the curated list of data. In an abstract sense, the model would be trying to figure out what is it that leads to lower or higher connectivity/visibility values across the different floorplans. When simulation engines are used, an analytical solution is produced. For the spatial analysis, for example, a grid is overlaid on the floor plan, from which a graph is constructed where each grid cell represents a graph node. The connectivity values on the floorplan are derived from how well each node is connected to all the other nodes. But the ML system does not have knowledge of that process. All it has is the input as an image comprised of black (obstacles) and white (unobstructed space) pixels and the output as an image where the white pixels are transformed into a greyscale gradient representing the spatial or visual connectivity.

During training, the model keeps trying to answer the above question. Its answers are terribly wrong at the beginning, but as time goes by and through the learning process of an ANN that is dependent on continuous feedback, the ANN starts to correct its mistakes and the answers start getting closer to being correct. At the end, the model finds its own recipe to map from input to output—and it is not one that the user provided, but one the model devised from the examples, by trial and error. Again, the objective of the training is to reach a state where the model can

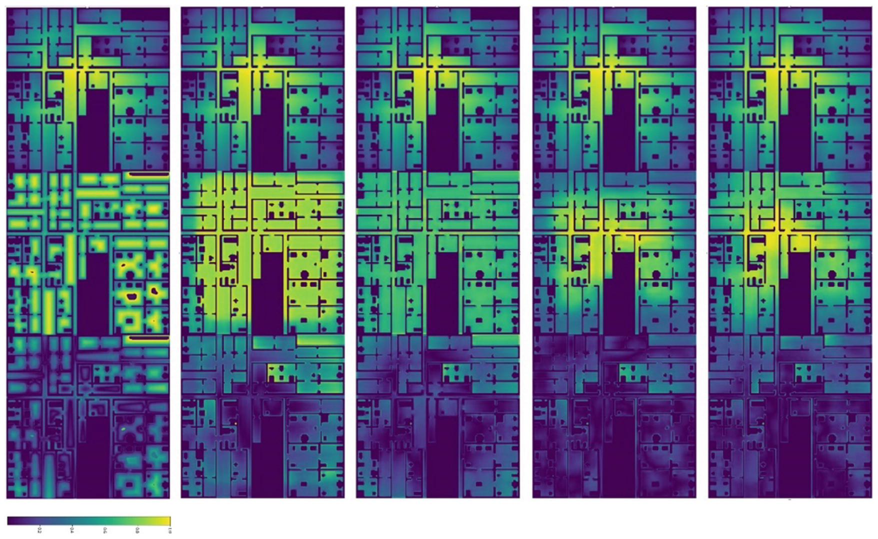

Fig. 14 Automatically generated floorplates and their respective analyses for the connectivity study. Progress of GAN-based network learning to predict connectivity analysis

provide the right answer for almost all the images in that dataset—a model that can generalize well and this system, as described, delivers just that.

For this study, we were comparing two different types of networks, one based on GANs, and the other is a convolutional neural network called U-Net [44]. Information about the data processing aspect of this study, the reasoning behind the choice of networks, the optimization of the networks' parameters and more is provided in [45].

Once the ML model is trained, it can be used to run both spatial and visual analyses on images of floorplans of various size. The time it takes to derive to the result is less than 0.03 s.

In Fig. 14, a snapshot of this process can be seen. Each column represents a snapshot in time during the model's training phase. At the top of each column we see one of the synthesized square floor plans with the connectivity analysis output overlaid on it. In the middle looking from left to right, we see the progress of the model trying to answer how the analysis should look like. At the bottom we see the difference between the real analysis (top) and the model's output (middle), the darker the whole image is, the closer the model's answer is to reality. Which we can see at the bottom of the far right column.

5 Looking to the Future

Given the rapid development and integration of ML, it is possible that in the next few years, a lot of the simulation analyses that designers rely on in their early stage design process will be replaced with surrogate models. This will allow for even faster design iterations than currently available and facilitate integration in design mediums leveraging augmented and virtual reality in mobiles, head-mounted displays and web applications. It will, in turn, speed up tasks that use evolutionary algorithms coupled with analytical performance analysis, producing vast datasets of information. It will push AECO to rely more on cloud-based services as means to access the inference capabilities of those models. Whether those capabilities will be integrated into different client applications or through web-apps is yet to be seen, although a lot of new products are adapting and pushing for the latter. This last point raises a lot of concerns about data privacy, especially in this industry, which makes research focusing on federated learning or privacy preserving learning very appealing.

Foster + Partners' current research is targeted towards design-assist models. Therefore, most efforts are concentrated in showcasing the possibilities that ML can open in AECO as a real-time assistant during the design process. The Applied Research and Development group is continuously working on identifying interesting design problems, without analytical solutions, for which there is qualitative and quantitative original data to train ML systems on.

In the next couple of years, design and construction firms will be in full retrospect about the amount of data they have been generating for years. Different initiatives will arise looking into means of gathering, standardizing, tagging and augmenting data in the AECO industry. Given the competitiveness of AECO, it is anticipated that this new corpus will be fragmented, rather than a holistic representation of where the industry is.

AECO companies will realize soon enough how integral it is to develop data pipelines that they can have control of and organize them in a way useful to ML or data-driven workflows. Roles related to data administration, analysis, governance and management will be in demand and should be positioned in away allowing them to drive and alter business decisions. Pursuing the goal of data acquisition, companies and new start-ups will consider the appeal provided by vertical integration or exploring different contracts that will allow data sharing partnerships.

Competition will not only be between large firms monopolizing AECO data as a by-product of their scale. New regulations for data sharing practises will come into play, allowing and encouraging smaller companies to create data alliances.

Architectural schools will further focus on the business and data aspect of architectural projects, balancing the objective and subjective sides of design. Hopefully, awareness will also be raised about issues of bias and intellectual property concerns and how those could affect the products of this industry.

In Foster + Partners, and specifically in the Applied Research and Development group, there is a lot of focus not only on the creative design process, but simultaneously on the many innovative ways by which it can be facilitated. That is why, new technologies are perceived not as a threat, but as a possibility; a possibility to explore new ideas, to enhance creativity and to constantly optimize and push the boundaries of design. A possibility to make a difference and to make the future come quicker for everyone.

References

1. Mitchell, T.M.: Machine Learning, 1st edn. McGraw-Hill Inc., USA (1997)
2. Alpaydin, E.: Introduction to Machine Learning, 2nd edn. The MIT Press (2010)
3. Ahirwar, K.: Everything You Need to Know About Neural Networks (2017). https://hackernoon.com/everything-you-need-to-know-about-neural-networks-8988c3ee4491
4. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. Science (80-)313, 504–507 (2006). https://doi.org/10.1126/science.1127647
5. Bishop, C.M.: Pattern Recognition and Machine Learning (Information Science and Statistics). Springer-Verlag, Berlin, Heidelberg (2006)
6. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach, 3rd edn. Prentice Hall Press, USA (2009)
7. Murphy, K.P.: Machine Learning: A Probabilistic Perspective. The MIT Press (2012)
8. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. The MIT Press (2016)
9. Hastie, T., Tibshirani., R., Friedman, J.: The Elements of Statistical Learning. Springer New York Inc., New York, NY, USA (2001)
10. Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. In: Proceedings of the IEEE, pp. 2278–2324 (1998)
11. Krizhevsky, A., Sutskever, I., Geoffrey, E.H.: ImageNet classification with deep convolutional neural networks. Adv Neural Inf Process Syst **25**, 1–9 (2012). https://doi.org/10.1109/5.726791
12. Brown, T.B., Mann, B., Ryder, N., et al.: Language Models are Few-Shot Learners (2020)
13. Huang, C.-Z.A., Vaswani, A., Uszkoreit, J., et al.: Music Transformer (2018)
14. Zhang, Z., Cui, P., Zhu, W.: Deep Learning on Graphs: A Survey (2020)
15. Qi, C.R., Su, H., Niessner, M., et al.: Volumetric and Multi-View CNNs for Object Classification on 3D Data (2016)
16. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation (2016). CoRR abs/1612.00593
17. Hanocka, R., Hertz, A., Fish, N., et al.: MeshCNN. ACM Trans Graph **38**, 1–12 (2019). https://doi.org/10.1145/3306346.3322959
18. Xu, Q., Wang, W., Ceylan, D., et al.: DISN: Deep Implicit Surface Network for High-Quality Single-View 3D Reconstruction (2019)
19. Aubret, A., Matignon, L., Hassas, S.: A Survey on Intrinsic Motivation in Reinforcement Learning (2019). ArXiv abs/1908.0
20. Furbush, J.: Machine Learning: A Quick and Simple Definition (2018). https://www.oreilly.com/content/machine-learning-a-quick-and-simple-definition/
21. T2D2.: Thornt. Tomasetti. https://t2d2.ai/
22. Si, J., Barto, A.G., Powell, W.B., Wunsch, D.: Reinforcement learning and its relationship to supervised learning. In: Handbook of Learning and Approximate Dynamic Programming, pp. 45–63 (2004)
23. Burgess, M.: This is How Netflix's Secret Recommendation System Works. https://www.wired.co.uk/article/netflix-data-personalisation-watching
24. Gomez-Uribe, C.A., Hunt, N.: The Netflix recommender system: algorithms, business value, and innovation. ACM Trans. Manag. Inf. Syst. **6** (2016). https://doi.org/10.1145/2843948

25. Chentanez, N., Barto, A., Singh, S.: Intrinsically motivated reinforcement learning. In: Saul, L., Weiss, Y., Bottou, L. (eds.) Advances in Neural Information Processing Systems, pp. 1281–1288. MIT Press (2005)
26. Tiu, E.: The 'Ingredients' of Machine Learning Algorithms (2019). https://towardsdatascience.com/the-ingredients-of-machine-learning-algorithms-4d1ca9f5ceec
27. Intsite (2017). https://intsite.ai/
28. Built Robotics (2016). http://www.builtrobotics.com/
29. Galanos, T., Khean, N., Düring, S., Chronis, A.: Cil & Giraffe: Ai Wind Comfort on the Cloud. https://cities.ait.ac.at/site/index.php/2020/04/08/our-ai-models-at-work-introducing-our-partnership-with-giraffe-technologies-2/
30. Asterisk. Thornt. Tomasetti. http://core.thorntontomasetti.com/asterisk/
31. UpCodes. https://up.codes/features/ai
32. ArchiStar. https://archistar.ai/
33. Spacemaker AI. https://www.spacemakerai.com/
34. Tramèr, F., Zhang, F., Juels, A., et al.: Stealing Machine Learning Models via Prediction APIs (2016). CoRR abs/1609.0
35. Claburn, T.: How to Steal the Mind of an AI: Machine-Learning Models Vulnerable to Reverse Engineering (2016). https://www.theregister.com/2016/10/01/steal_this_brain/
36. Bonawitz, K., Eichner, H., Grieskamp, W., et al.: Towards Federated Learning at Scale: System Design (2019). CoRR abs/1902.01046
37. Openmined. https://www.openmined.org/
38. Davis, D., Agrawal, P.: Transparency around image cropping and changes to come. In: Twitter Blog (2020). https://blog.twitter.com/official/en_us/topics/product/2020/transparency-image-cropping.html
39. Dastin, J.: Amazon scraps secret AI recruiting tool that showed bias against women. In: Reuters (2018). https://www.reuters.com/article/us-amazon-com-jobs-automation-insight-idUSKCN1MK08G
40. Bender, E.M., Friedman, B.: Data statements for natural language processing: toward mitigating system bias and enabling better science. Trans. Assoc. Comput. Linguist. **6**, 587–604 (2018). https://doi.org/10.1162/tacl_a_00041
41. Olah, C., Satyanarayan, A., Johnson, I., et al.: The building blocks of interpretability. Distill (2018). https://doi.org/10.23915/distill.00010
42. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., et al.: Generative adversarial nets. In: Proceedings of the 27th International Conference on Neural Information Processing Systems, vol 2, pp. 2672–2680. MIT Press, Cambridge, MA, USA (2014)
43. Abdel-Rahman, A., Kosicki, M., Michalatos, P., Tsigkari, M.: Design of thermally deformable laminates using machine learning. In: Advances in Engineering Materials, Structures and Systems: Innovations, Mechanics and Applications, pp. 1016–1021. CRC Press (2019)
44. Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional Networks for Biomedical Image Segmentation (2015). arXiv e-prints arXiv:1505.04597
45. Tarabishy, S., Psarras, S., Kosicki, M., Tsigkari, M.: Deep learning surrogate models for spatial and visual connectivity. Int J Archit Comput **18**, 53–66 (2020). https://doi.org/10.1177/1478077119894483