# Chapter 16
# Major Technology 10: Artificial Intelligence (AI) in Virtual Product Creation


**Executive Summary**

This chapter deals with the following topics:

- Basics and advanced techniques of Artificial Intelligence in Virtual Product Creation (VPC)
- Providing insight into how engineers benefit from using Artificial Intelligence (AI) technologies in VPC
- Describing functioning, benefits, and limitations of AI technologies in VPC practice.


**Quick Reader Orientation and Motivation**

The intention of this chapter is:

- to give an overview of AI technology in Virtual Product Creation as driver and enablers for Digital Transformation in engineering
- to present AI technology as part of Virtual Product Creation from a practitioner's point of view to analyze the need and usefulness for day-to-day industrial work practice
- to give instructions on how to use AI technology
- to explain models, frameworks, and digital representations that help to grasp the internal working modes of AI technology in Virtual Product Creation.

*Artificial Intelligence* (AI) is not a new concept or technology. It first appeared in the 1950s, when several scientists came together with the dream to build machines as intelligent as humans. Afterwards, this field has experienced several hype cycles, including the so-called AI Winters, in which many research organization and companies failed to deliver their extravagant promises [1]. In the 80ties and 90ties of last century a wide variety of rule- and knowledge-based AI systems have already been introduced in industrial engineering work and in technical system operations (e.g. as part of damage analysis tools or engineering assistant systems in design synthesis). In the 2010s, the term of AI rose again, especially the sub-field of *machine learning*, due



R. Stark, *Virtual Product Creation in Industry*,
https://doi.org/10.1007/978-3-662-64301-3_16

to the development of next generation of computing power (e.g. Graphics Processing Unit (GPU), Clouding Computing, etc.), the increased amount and variety of data and the advances in algorithms, especially in *Deep Learning* (e.g. Artificial Neural Networks). Before delving into the definition of Artificial Intelligence, the definition of intelligence will be hereinafter introduced.

## 16.1 What is Intelligence? What is Artificial Intelligence?

**Intelligence** can be defined in many ways and that is why it may be controversial to try to find a unique comprehensive definition of the term [2]. From the perspective of psychologists, intelligence can be defined as the ability to solve problems, to create products that bring values within cultural settings [3]. From the perspective of AI researchers, it can be defined as the ability to process information properly in a certain environment. In order to define the criteria for an appropriate definition of intelligence, it is required that information is processed by corresponding experts [4].

Accordingly, AI owns a significant variety of subfields, ranging from general (learning) to specific tasks. Such as playing GO, writing lyrics, face detection, self-driving cars, diagnosing diseases, etc. Figure 16.1 [1] illustrates eight definitions

| | |
|---|---|
| **Thinking Humanly**<br><br>“The exciting new effort to make computers think … *machines with minds*, in the full and literal sense.“ (Haugeland, 1985)<br><br>“[The automation of] activities that we associate with human thinking, activities such as decision-making, problem solving, learning …” (Bellman, 1978) | **Thinking Rationally**<br><br>“The study of mental faculties through the use of computational models.“ (Charniak and McDermott, 1985)<br><br>“The study of the computations that make it possible to perceive, reason, and act.” (Winston, 1992) |
| **Acting Humanly**<br><br>“The art of creating machines that perform functions that require intelligence when performed by people.” (Kurzweil, 1990)<br><br>“The study of how to make computers do things at which, at the moment, people are better.” (Rich and Knight, 1991) | **Acting Rationally**<br><br>“Computational Intelligence is the study of the design of intelligent agents.” (Poole et al., 1998)<br><br>“AI … is concerned with intelligent behavior in artifacts.” (Nilsson, 1998) |

**Fig. 16.1** Artificial intelligence explanation, organized into four categories [1]

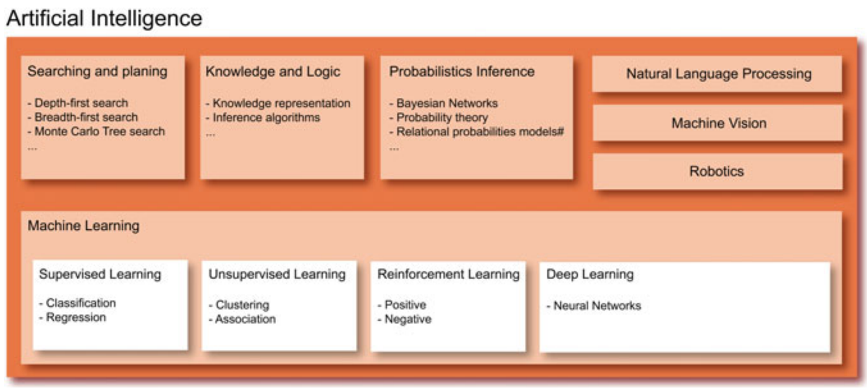



**Fig. 16.2** Categories of AI based on [1]

of AI, divided into two dimensions. The above definitions are about *thinking* and *reasoning*, whereas the ones below are about *behavior*. Definitions in the left column measure success of AI according to how they are similar to human behavior, whereas the ones on the right column concern about the ideal performance of AI systems, called *rationality* [1].

Research about AI has the initial objective to build machines that could help to improve our understanding of intelligence. The technologies of AI can be broadly divided into the following types [5, 6]:

- Knowledge-based systems: explicit modeling with words and symbols
- Computational intelligence: implicit modeling with numerical techniques.

With the renewed rise of AI in the 2010s, terms such as are Machine Learning, Deep Learning and Neural Networks also increased in popularity. However, AI is a much broader concept and consists of many more subfields than these ones. Figure 16.2 illustrates different categories of AI.

## 16.2 Knowledge-Based Systems and Their Application in Industry

Knowledge-based systems are designed to answer complex questions within specific domains. They include techniques such as rule-based, model-based and case-based reasoning. They were among the first forms of AI and remain in a major position until now. In the simplest case, knowledge-based systems contain three modules: knowledge base, inference engine and user interface (as in Fig. 16.3). In knowledge base, the declarative description of problems is stored, e.g. some rules, facts and relationships, without the details about *how* or *when* to apply them. These details exist in inference engine. Since knowledge is explicitly described in knowledge base,

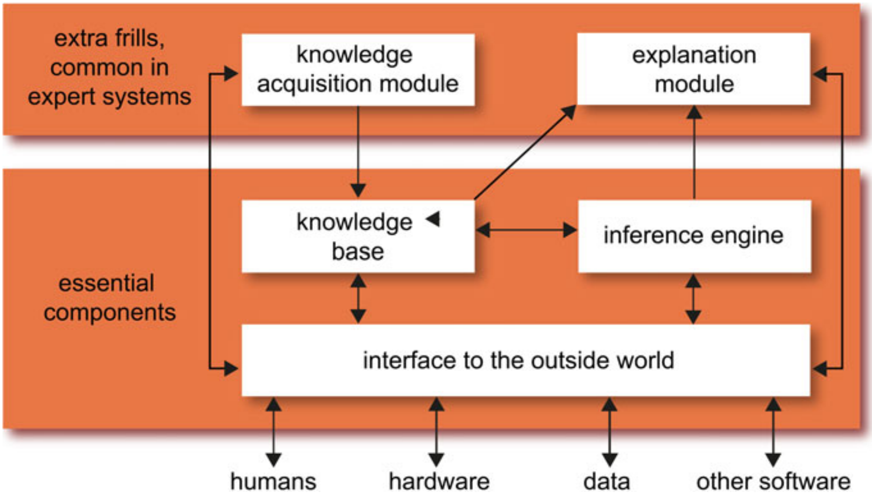



**Fig. 16.3** The main components of a knowledge-based system [5]

rather than implicitly embedded in the structure of a program, domain experts can relatively easy update knowledge without any programming skills [5]. An example that displays the problem explicitly is "*If the pressure is high and the release valve is closed, then the release valve is stuck*" [6] On the other hand, the way an inference engine uses knowledge base is similar to the way a conventional program calls a data file [5].

Expert system is one type of knowledge-based system, which is designed to integrate the expertise into a specific domain, such as medical diagnoses and technical diagnoses. It is intended to act as a human consultant that could offer answers according to their domain expertise. Normally, the user interacts with the expert system by describing the problem through dialogues. Then the expert system offers answers, suggestions, or recommendations. Typically, the expert system shall be able to justify the current line of inquiry and explain the reasoning of conclusion, and this is the function of explanation module in Fig. 16.3 [5].

Knowledge based systems are one of the first AI applications created. In 1969, a program called *DENDRAL* [7] was initiated by Ed Feigenbaum, Bruce Buchanan and Joshua Lederberg, with the purpose to deduce molecular structure using information provided by a mass spectrometer. It was also the first expert system, written in programing language LISP,[1] which automated decision making and problem solving processes for chemists. It reached significant success at that time, since it clearly separated rule-based knowledge from reasoning component, which mapped knowledge from a general form to special forms, like cookbook recipes [5]. With lessons learned from DENDRAl and the objective to prove, the methodology of expert systems could

[1] LISP: short for List Processing, a favored programming language for artificial intelligence, which is based on lambda calculus. Works good for computation associated problems.

also be applied to other sort of human expertise like MYCIN, which was developed by Ed Feigenbaum, Bruch Buchanan and Dr. Edward Shortliffe to diagnose blood infection. MYCIN was able to perform as well as some experts, and even better than junior doctors. With the growth of applications of expert systems with the aim of facing real world problems, different representation and reasoning languages were developed, e.g. Prolog[2] [5].

With the success of commercial expert systems, there was an AI boom during 1980s and 1990s in which companies invested millions to billions of dollars in building expert systems, vision systems, software and hardware to implement AI systems. This success raised great optimism for AI, but only for applications for specific narrow domains. Applications for more broad-based representations of human intelligence were still difficult to achieve [8]. Typical rule and knowledge-based AI applications ("first generation AI industrial applications") were introduced to support the following tasks and application fields back then and are still effective today:

- Failure and damage analysis and explanation (reasoning).
- Model design synthesis and concept classification.
- Design knowledge templates to support design automation.
- Checking routines in engineering design and release as well as in technical system maintenance and overhaul.
- Business case calculation, cost estimation and financial assessment.

## 16.3 Machine Learning—The Most Widely Used AI Subfield in Industry

In this sub-chapter, the author focuses on Machine Learning, since it is currently the most widely used category of AI in industry. However, before approaching this topic further, some basic Machine Learning (ML) terminologies are in the following explained:

- **Attribute** [8]: also known as an independent variable or feature, which describes an observation (e.g. height, color, etc.). Generally, attributes are divided into the following two types:
  - **Categorical**: discrete values, which can be divided into two subtypes: *nominal*, in which there is no ordering between the values, such as last names and colors; *ordinal*, in which there exists an ordering, such as low, medium or high.
  - **Continuous** (quantitative): subset of real numbers, which means there is measureable difference between values.

[2] Prolog: a logic programming language, which is widely used in artificial intelligence and computational linguistics. It works well for rule-based logical queries.

- **Hyperparameter**: a high-level property of an AI model, which decides the learning rate and complexity of the model.
- **Model** [8]: also known as *classifier*, it is a structure or interpretation, which summarizes or partially summarizes a set of data, for description or prediction purpose. The result of most AI algorithms is such kind of models.
- **Knowledge discovery** [8]: the process to identify valid, novel, potential, useful and understandable patterns in data. This concept was first used in "Advances in Knowledge Discovery and Data Mining", 1996, by Fayyad et al. [9].
- **Training data**: the subset of data, which is used to observe, to learn and to train a model.
- **Test data**: the subset of data that is used to test the performance of a model after the model has been trained with training data and validated with validation data.
- **Validation data**: the subset of data apart from the training data, which is used to adjust the hyperparameters of a model.

*Learning* is the process in which the AI System is able to improve its performance on future tasks after making observations about the world [1]. *Machine Learning* is a subfield of AI that became an extremely popular term in the last decade. It is the science to get computers/programs to learn from experiences rather than programming with specific rules. A detailed definition from Tom Mitchell reads as follows [10]:

*A computer program is said to learn from* ***experience E*** *with respect to some class of* ***tasks T*** *and* ***performance measure P*** *if its performance at tasks in T, as measured by P, improves with* ***experience E***.

This definition of *Machine Learning* also defines the general guidelines to start any new projects in this field: before starting any *Machine Learning* project, the task (objective) T, the performance measure P and the experience E should be defined.

According to different learning styles, *Machine Learning* (ML) could be grouped into the following four types:

1. **Supervised learning**: the training data fed to the algorithm is labeled, i.e. the samples are marked or augmented with a meaningful tag which represents information. The algorithm learns the relationship (a function), which maps the given input data to the given output data [1]. According to the output types, supervised learning can further be grouped into *regression* and *classification* problems. In *regression* problems, the output is a continuous numerical value, such as 'weight' of a constructive part. For instance, in the absence of an analytical equation, if the radius of a part is 4.9 cm and the weight is *predicted* to be 200 g, then 200 g is the output of a *Machine Learning* system. Figure 16.4a illustrates a regression problem. In *classification* problems, the output is one of the labels/categories of the input dataset. Figure 16.4b illustrates a classification problem.

2. **Unsupervised learning**: the training data fed into the algorithms is *unlabeled*, i.e. no additional information exists for the samples. Algorithms learn the pattern

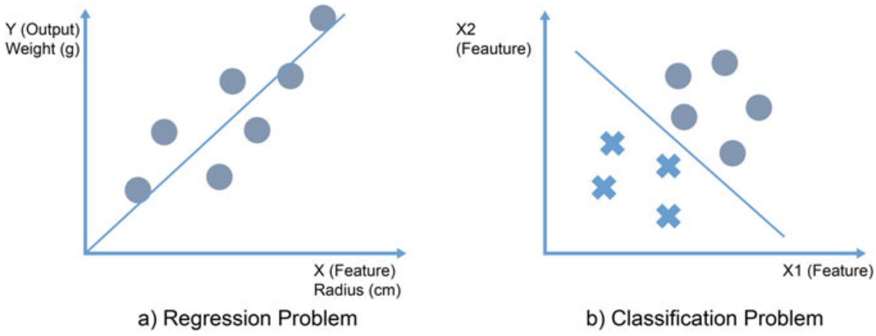


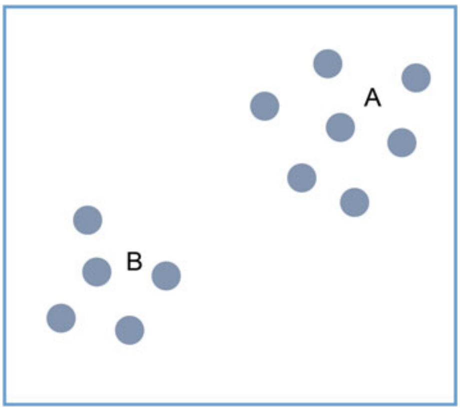

**Fig. 16.4** Supervised learning

in the training data without being given an explicit output and model the underlying structure or distribution in the data. According to the *output types*, unsupervised learning can be divided into *clustering* and *association* tasks. In *clustering* tasks, the main focus is to detect potential useful clusters of the input sample [1]. Figure 16.5 is an illustration of a clustering problem. In *association* tasks, the main focus is to discover rules which describe the large amount of the training data.

3. **Semi-supervised learning**: In practice, the differences between supervised and unsupervised learning are not so obvious. In semi-supervised learning, the input is a mixture of labeled and (a lot of) unlabeled data. And even the labeled data may not be 100% correct [1]. Unlabeled data is easier to acquire, compared with labeled data, and the labels may require support from experts or special devices/software.

   The most common application about semi-supervised learning is the photo hosting services, such as Google Photos and Apple iOS Photo Stream: when photos are uploaded to the service, it automatically recognizes that the same

**Fig. 16.5** Unsupervised learning—clustering problem



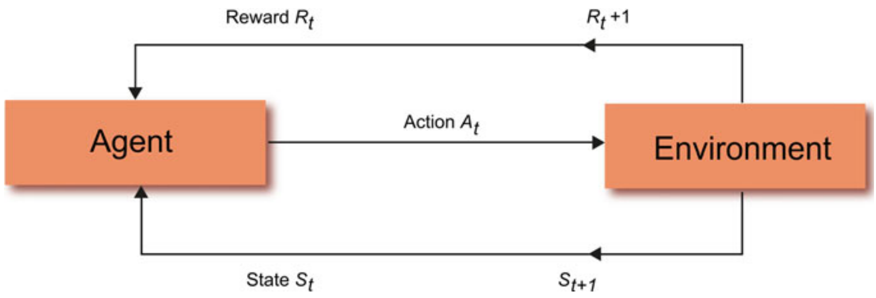



**Fig. 16.6** The agent-environment interaction in reinforcement learning based on [12]

person A appears in photo #1, #3, #5 and person B in photo #2, #4, #6. This is also an unsupervised learning problem—clustering. If the system is informed about who these persons are, just by means of being labeled in one of the photos, then the system is able to name everyone in every photo. The same applies if pictures are taken from cracks of mechanical structures and certain types of them are categorized towards certain types of failures such as coating hairline crack, full surface crack or substantial (volumetric) fracture.

4. **Reinforcement learning**: the learning system makes *observations* in an *environment*, takes *actions* and in return, receives *rewards*. The learning system must learn the best strategy by itself by maximizing the rewards, this is called a *policy*. A policy describes what action the agent should choose in a given situation [11]. When there is no sufficient training data or the only way to learn about the environment is to interact with it (i.e. the ideal state is not clear), then reinforcement learning could play the biggest role. Figure 16.6 is an illustration of the agent-environment interaction of reinforcement learning.

    A simple example is a robot (agent), which applies reinforcement learning to learn to walk in a case when there exist two routes in front, a route A with fire and another route B with water. It firstly observes the environment and constructs its own representation of the environment (state), then it takes an action. If it chooses route A, it will get burned (next state) and will get negative reward. Then, it knows it should take fewer actions that lead to such a result (updating policy). On the other hand, if it chooses route B, it will get positive reward and it knows it should take more actions that lead to the result in the future. The robot will repeat the process until it finds a policy (what to react to under different circumstances), which maximizes the rewards.

Similarly, in manufacturing, the Japanese company *Fanuc* [13] has applied reinforcement learning to improve the efficiency and precision of industrial robots. A robot learns to train itself by picking up objects (*actions*) while capturing video footage of the process. After every success or failure, it records how the object looked like and all the relevant features, which are the *state* of the process. The robot gets a positive reward when it puts the parts into the correct container; otherwise, it gets a negative *reward*. The goal is to come up with a *policy* which tells the robot, which kind of part should be put into which container.

### 16.3.1 Deep Learning

*Deep Learning* is a subset of machine learning, which can also be divided into the learning types of unsupervised, supervised, semi-supervised and reinforcement learning. The major difference between standard *machine learning* and *deep learning* is that in standard *machine learning* the training data is described by a set of fixed-length features or attributes, whereas the features or attributes are to be extracted from the *raw input data* in *deep learning*. In other words, *deep learning* can process a large amount of data and at the same time requires less *data preprocessing* time. This is accomplished by utilizing one to many interconnected layers (hidden layers) of calculators, an input layer and an output layer, which form a basic structure of *neural network* (Fig. 16.7). This architecture is inspired by the brain, which is why the calculators are also known as 'neurons'.

The input layer of a *neural network* processes a large amount of raw input data. Then the hidden layer(s) in between learn(s) to increase the details of input features. The output layer is responsible for making a determination about the input data and afterwards, when the neural network is applied to new input data, it will make a prediction based on what it has learned. For instance, in order to recognize if the same person has appeared in the new picture.

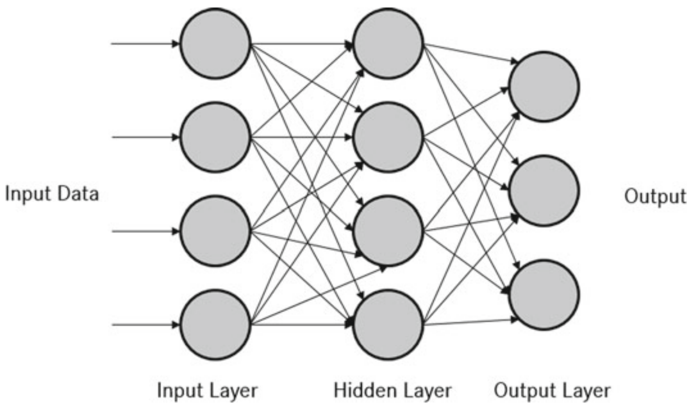



**Fig. 16.7** A simple architecture of neural network

### 16.3.2 Standard Process for Machine Learning Projects

*Machine Learning* itself is just a core step of the complete methodology to deal with AI projects. Without a deep understanding of the existing problem and available data, it is difficult to achieve the objectives.

With the promotion of Industrie 4.0, comprehensive data is collected in companies within connected machines and systems. According to the *Wissenschaftliche Gesellschaft für Produktionstechnik* (WGP) in 2019 [14], the innovation and competitiveness of manufacturing companies is based to a large extent on the technological knowledge of engineering processes, machines and systems. The key question is how to link the knowledge with the new development of AI systematically and methodically in order to increase the efficiency and added value of processes, machines and plants in addition to the value creation in engineering [14]. There are in general two kinds of approaches to apply AI in the area of engineering: *data-driven* and *process-driven.*

For the *data-driven* approach, companies first collect a large amount of data by applying data analytics to find useful information from it. This relies more on an information technology perspective, which does not require much knowledge with respect to engineering processes. The disadvantage is that normally the collected data is not gathered consequently to existing engineering processes. Therefore, only limited possibilities to get valuable information from the data with respect to the specific engineering process steps are available [14].

The *process-driven* approach is generally aimed at monitoring, controlling or optimizing the process. It highly depends on the type of steps, the machines, the environment, the material and the people involved in the process. Therefore, in order to answer the questions of which data are needed and how to collect such data, an extensive knowledge on the domain is required and such knowledge has a high influence on the results of the AI project. Compared to the *data-driven* approach, the *process-driven* approach systematically extracts more valuable knowledge from engineering processes [14]. In Fig. 16.8, a new value creation potential is shown in the form of 'sensorisation'. The learned model can be applied to optimize the complex process, to evaluate the reliability of results and to improve the visualization the results for better decision making [14].

*Cross-Industry Standard Process for Data Mining* (CRISP-DM) [16] is the most popular model designed to orient researchers for a standard AI Project. It fits for both, *data-driven* and *process-driven* approaches, which are already mentioned above. The

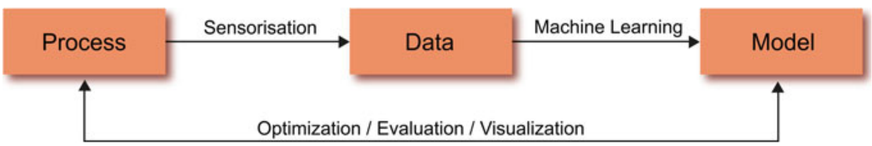



**Fig. 16.8** Production technical process-driven approach to apply machine learning [14, 15]

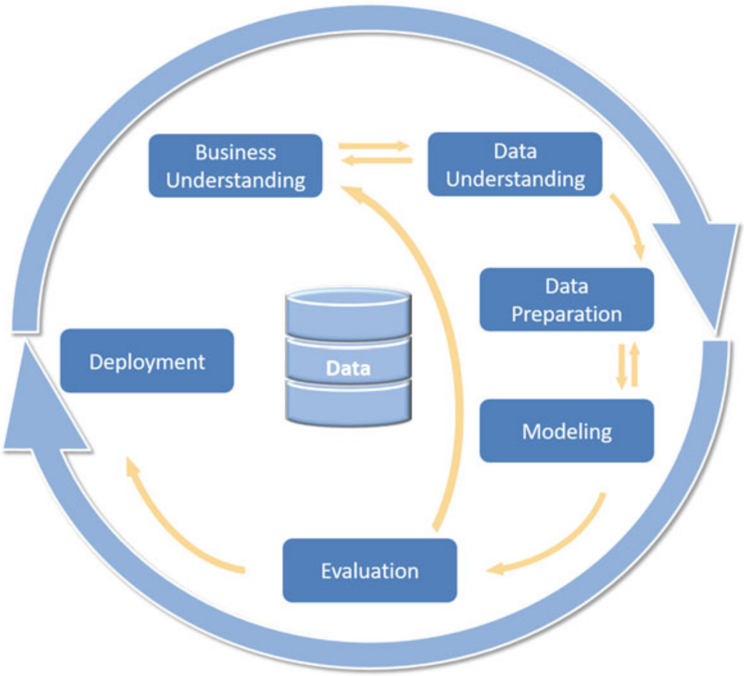



**Fig. 16.9** The six phases of the CRISP-DM process [16]

value of CRISP-DM lies in that it involves data science steps that range from business needs to deployment and indicate how interactive the process is. CRISP-DM (see Fig. 16.9) in general describes six phases:

**Business understanding**: this step begins with an enterprise/industrial or academic need for learning new knowledge or improving current processes. Then the objective is defined, followed by assessing the current situation that needs technical/process knowledge. Afterwards, a plan for finding such knowledge is defined, such as how to collect data, analyze and report data. And then it will be transformed into the objective of the AI project [17]. For example, an automotive company wants to improve the quality and efficiency of constructive design work in CAD system by developing a CAD assistant system. Goals such as "According to the engineers' current design work, what are the next best *features* to be used?" "What kind of *parameters* should be chosen?" are needed.

**Data understanding**: after a clear description of the problem, the relevant data should be identified and collected. This is followed by exploration and quality assessment of the data [17]. For instance, in CAD systems, there are *log files* which record the used command (feature) combined with parameters and default geometries. According to the logged history, the design behavior can be learned by means of AI algorithms.

**Data preparation**: the purpose of this step is to select and clean the required data for better quality. This includes: integration, filtering of outliers, filling the missing values, etc. Besides, the selected data may also have different formats [17]. For instance, the CAD system log files need to be converted to a readable format for AI algorithms, e.g. csv. The feature name sequences need to be converted to integers in order to fit AI algorithms.

**Modeling**: depending on the problem, the appropriate modeling techniques will be selected. Different techniques could be applied in this step, results will be compared and the most appropriate technique will be decided. In the example of CAD assistant system, algorithms like *Random Forest*[3] and *Multilayer Perceptron (MLP)*[4] are applied and compared in the beginning, and *MLP* is chosen due to a better performance.

**Evaluation**: the results of modeling need to fit the business purpose and should be evaluated in context of business success criteria. In this step, an interaction of data analyst, business analyst or (virtual) engineering experts and decision makers is mandatory [17]. For instance, for evaluation of the CAD assistant system, design engineers, data analysts and managers get involved to assess the degree to which the model meets the business objectives.

**Deployment**: the results of modeling will be distributed as a usable representation and integrated in an organization process/system. In the example of a CAD assistant system, it is integrated into a CAD system such as *Siemens NX* or Dassault Système CATIA as a plugin and running in parallel with it. The performance will not be mutually affected.

## 16.4 (Big) Data in Product Lifecycle Management

To achieve better performance in *Product Lifecycle Management (PLM)* [18] with the support of *Big Data* and *Artificial Intelligence*, it is necessary to clarify which type of data sets are involved in which phase of the lifecycle of a given product, machine (both represent technical systems) or service. Generally, the product lifecycle could be divided into: Begin of Life (BOL), Middle of Life (MOL) and the End of Life (EOL) [9]. In BOL, the product concept is generated, designed and physically tested and its production is being prepared. In MOL, products are produced, distributed, used and maintained by customers or engineers. In EOL, products are prepared for re-use and/or recycled by manufactures or disposed by customers [19, 20].

**Begin of life, BOL**: According to Jun et al. [20], the most essential steps involved in BOL are: market analysis, product design and production preparation. In phase

[3] The algorithm *Random Forest* is based on a combination of decision trees. To classify a data sample, each decision tree provides a classification result for the input data. *Random Forest* then collects the results from each decision tree and choose the most voted one as the prediction result [18].

[4] *Multilayer Perceptron* (MLP) is the simplest neural network, sometimes also referred to a feedforward Artificial Neural Network.

market analysis, the target is to meet customers demand. There exist a variety of data formats, e.g. comments on blogs, videos that customers upload on the Internet, websites in which customers mark their purchasing behaviors. Besides, the information from MOL and EOL, for instance, customers' complaints and sales performance of similar products can also contribute to provide the goals for product design [19].

In the phase of product design/development and manufacturing engineering, the data involved can be the descriptions of needs, requirements, description of specific product functions, detailed design specifications—e.g. drawings or product configurations, the accurate programming codes for the automation of manufacturing equipment, and all kinds of technical parameters. Furthermore, the maintenance and failure information from MOL, like the records of breakdowns and root causes can also contribute to efficient and reliable product design [19] as part of "feedback-to-design".

**Mid of life, MoL**: In the middle of the product lifecycle, the product exists in its final form. The main issues and influence factors can come from production and from service [19].

In the production phase, while some data might be stable, other data are dynamic and change along the phase of product manufacturing. The data from product design will be regarded as standards for production processes and operation, and data from monitoring and testing of products are used to check whether all standards are reached and met [19].

In the logistics phase, warehouse management and transportation need efficient decision strategies to solve complex issues. Based on the order information, here considered as input data, the manufactures are able to find optimal arrangements. One of the main tasks along this line is to transfer order information into intelligent arrangement within a global view and supply chain network [19].

In the utility phase, customers operate products based on the information from user manuals or from heuristic knowledge. In this process, product status information are generated and potentially transferred back to manufactures: traditionally, for most of the products in field usage only failure modes are recorded, nowadays, due to internet technologies, the actual (positive) use data become decisive for new business models of manufactures during the utility phase. In addition, the field usage information is monitored and recorded to provide guidance for the product maintenance [19]. In the maintenance phase, by combining maintenance supporting information with product status information generated from utility phase, faults can be predicted and prevented. The adjusted maintenance plan with root causes and solutions is taken into account as output data during this phase [19].

**End of Life, EoL**: In the end phase of product lifecycle, lots of decisions have to be made regarding EOL product re-use (or partial re-use), recycle or disposal. With the help of data from MOL the following decisions can be supported: maintenance history information, product status information and usage environment information, the degradation status and calculation of remaining value of individual components. The purpose in EOL is to maximize values of products. Depending on the status of the product, suitable options such as recycle, re-use, remanufacturing, and disposal should be decided [19].

## 16.5 Internet of Things

The majority of current industrial products are mechatronic. With the evolution of micro embedded devices and software within mechatronic products, their intelligent capabilities, such as autonomy, real-time interaction, self-organization, etc. and the capabilities to communicate and network with other products have been improved. This type of product is now defined as 'cyber-physical systems (CPS)' [21].

The term "*Internet of Things (IoT)*" was first suggested by Kevin Ashton [22] in 1999. At that time, he viewed Radio-frequency identification (RFID) as the essential to the internet of things. Literally, IoT means "*…all about physical items talking to each other …*" [23]. Nowadays, IoT carries a much broader designation since the term IoT is oftentimes also referred as a term to describe daily used gadgets and objects with internet connection such as TVs, smart watches, cellphones, ovens, refrigerators, cars, etc. All of them, however, handle data sets created by sensors in those objects and gadgets of daily live as well as in machines of smart factories.

Making products 'smart' means connecting and sharing data between them. On the other hand, it means capturing the huge amount of data, ingest, process it and then mine it as the business requires. Enabled by IoT, CPS could not only communicate and network with each other, but are also capable to perform a required functionality by integrating the available internet services. These products are called '*Smart Products*' [21].

There are many design challenges faced by the developer and engineers of smart products. Among many issues, such as availability of internet, the IoT is entirely dependent on the development of Wireless Sensor Networks (WSN) and Radio Frequency Identification Devices (RFID). Mukhopadhyay [23] has summarized the many challenges of IoT as follows:

- Availability of Internet everywhere and at no cost
- Security issues
- Low-cost smart sensing system development
- Energy
- Computational ability
- Scalability
- Fault Tolerance
- Power Consumption.

In 2014, a framework of CPS was proposed by Lee et al. [24], which provides a guideline for applying CPS to industrial use cases. This architecture consists of 5 "C"-levels:

- **Connection**: this level consists of properly selecting sensors and data sources, transferring protocols, and seamlessly transferring data to the central server [25].
- **Conversion**: in this level, intelligent algorithms and data mining techniques can be applied to various raw data to extract valuable information, which is also known as features in most Machine Learning projects. Then the calculated information along

with other machine state data is being sent through Ethernet or Wi-Fi Network to a cloud server, in which the information is managed and stored [25].

- **Cyber**: information from every connected machine will be gathered and analyzed in this level. The results of the analytics provides machines with self-comparison ability—performance of the individual machine can be compared and rated among the fleet [24] and with historical information of similar machines to predict the future behavior of this machine.
- **Cognition**: in this level, the acquired knowledge is presented as comparative information as well as individual machine status to experts, in order to support better decision making. Therefore, information visualization techniques such as graphics, tables are necessary to transfer the acquired knowledge completely [24].
- **Configuration**: this level acts as the feedback from cyber space to physical space. It applies the corrective and preventive decisions that have previously been made to the cognition level to the monitored system [24].

**Case study: Cyber-physical system-based smart machine**

So far, the application of AI based algorithms have become popular in real physical application cases, such as in manufacturing. The following case study of the "sawing material" example explains the approach and the appropriate measures which are necessary to apply the five "C"-level approach of Lee. Manufacturing processes start with sawing raw materials into designed sizes, therefore, speed and quality of sawing affect the whole production. Errors in sawing will propagate to the following steps and further affect the quality of product. Accurate sawing requires slowly cut but since it will affect the productivity of the production, an optimal balance between quality and speed need to be achieved [25].

In the **connection** level, data is collected from sensors and controller signals. Data, such as vibration, acoustic emission, temperature, blade speed, cutting time and blade height, etc., provide working status of each machine and will be processed in the industrial computer connected to each machine [25].

In the **conversion** level, the industrial computer performs feature extraction and data preparation. For instance, frequency domain features such as RMS (Root Mean Square), kurtosis, frequency band energy percentage, etc. are extracted from vibration and acoustic signals. At this stage, however, it is crucial to use manufacturing processing know-how and process knowledge (compare Fig. 7.10.8). Calculated features together with machine state data are sent through Wi-Fi network or Ethernet to the cloud server for storage and management [25].

In the **cyber** level, an adaptive clustering method [26] is performed on the cloud server to segment the historical performance of blades into discrete working regimes based on the difference of features comparing to normal baseline and local noise distribution. The clustering method (see explanations to unsupervised learning and Fig. 16.5) then compares the current features with the baseline and historical working regimes and identifies the appropriate cluster to match with the current working condition. If no appropriate cluster is found, a new cluster is generated [25].

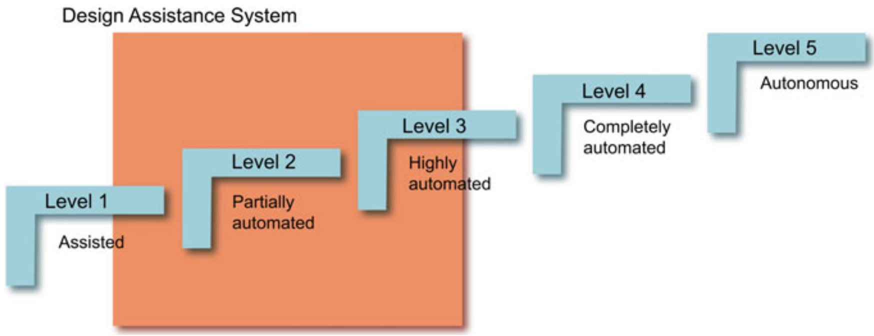



**Fig. 16.10** The vision and levels of design assistance systems at Daimler (based on Daimler internal project material)

In the **cognition** and **configuration** level, decisions will be made based on the health information of each connected machine, which is visualized by Web or iOS-based user interface. For instance, for a new blade, a higher cutting speed will be chosen for high productivity without affecting the quality of production. After a certain amount of degradation, a more moderate cutting should be applied to ensure production quality [25]. Please refer to Chap. 20 for more details on "*Internet of Things (IoT)*", especially with regards to "*Industrie 4.0*".

## 16.6 Example of a Virtual Product Creation AI Application

Industry leaders have recognized the widespread of digitalization. Instead of been changed by the digital wave, many companies have decided to react to changes and be a game changer by implementing new technologies supported by agile working methods. AI is definitely one of the promising technologies which helps companies on the way to digital transformation. And it has a place in the future of Computer-Aided Design (CAD), as one AI example in Virtual Product Creation.

Recently, Daimler AG has developed a design assistant system—NeuroCAD[5] with the objective to support CAD Data construction in Siemens NX CAD System by means of *Artificial Neural Networks*, which are a sub-discipline of Artificial Intelligence. Similar to the five levels of autonomous driving, the vision of NeuroCAD is to enable highly automated design (see Fig. 16.10). There exist three assistance systems components of NeuroCAD: the (design) feature assistant, the structure assistant and the parameter assistant. Meanwhile, NeuroCAD has reached the capability to partially automate the CAD design work, which goes beyond the "high end template based" design automation approach from the first and second decade of this millennium.

[5] NeuroCAD is a separate program which runs in parallel with Siemens NX. The performance will not be influenced mutually.

### 16.6.1 *The Main Function Description*

This sub-chapter explains the main functional elements of the design assistance system NeuroCAD, which employs a range of AI elements.

**Feature assistant**: this first functional element learns the typical command sequences (NX functionalities) and then supports design engineers by selecting the next best commands in NX by suggesting the three most likely commands the user could use next (as shown in Fig. 16.11). Each click will recall the corresponding command in NX. If all suggested commands are not appropriate, the user can still choose commands directly in NX. Feature Assistant in this case, provides only suggestions as part of design assistance instead of automating the design work.

**Structure assistant**: the second functional element is the traditional feature-based modelling (compare the sub-chapter "Feature based Modeling", part of the Chap. 7 "Computer Aided Design—CAD"). The CAD System (in this case *Siemens NX*) keeps the history of each command (feature) with the used parameters and the default geometries in a structure tree as part of the traditional CSG based modeling paradigm (compare Chap. 7 "Computer Aided Design – CAD").

Each feature can be modified later and all subsequent features of the design will be recalculated. These geometry construction features build high interconnectivity of data. It is difficult to master the complexity if the data is not further structured. Designers thus use '*Feature Group*' function in NX to group the features applied to a specific geometry. Some typical constructive commands will be repeated with variants. Thus, the creation and extension of similar components can be suggested by using the historical logs and AI technology. The *Structure Assistant* can support in such a case: it makes suggestions during the creation of feature groups and recommends the appropriate feature groups based on the position of part in the Part-Navigator.

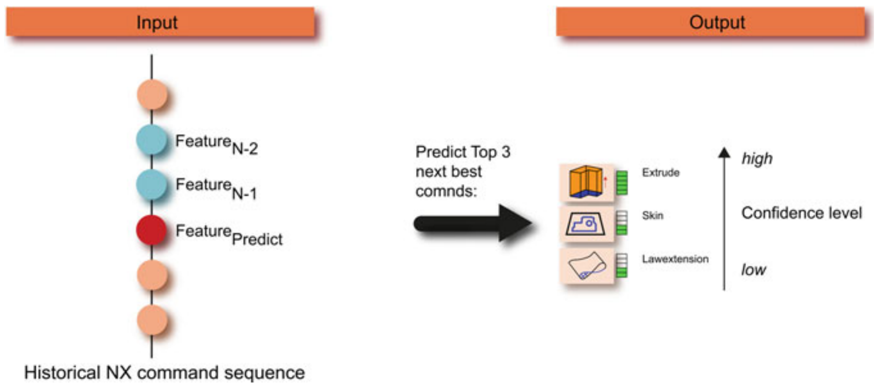



**Fig. 16.11** Feature Assistant: by learning from historical NX command sequence, the next best 3 commands are predicted and displayed in confidence level from high to low (on the right side)

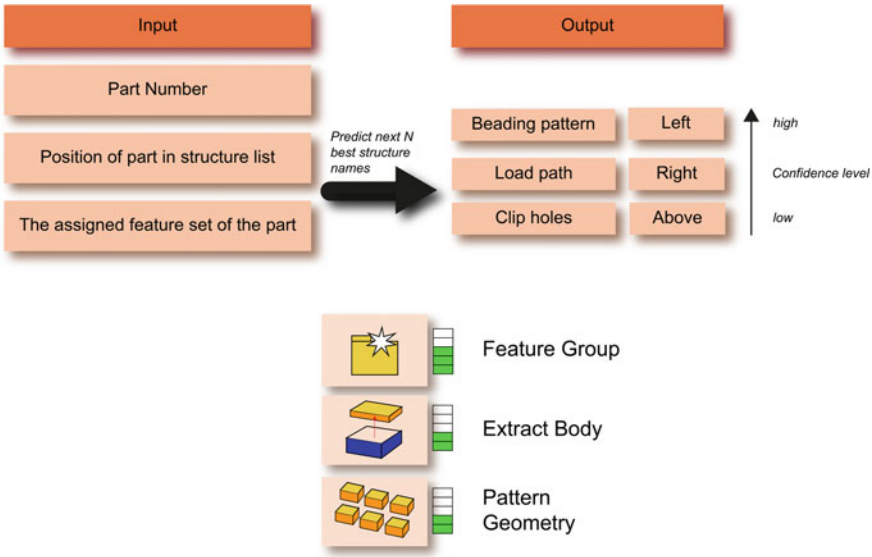



**Fig. 16.12** Structure Assistant: by learning from information about part (e.g. part number, position of part in structure list and its corresponding feature set), the next best structure names are predicted and displayed on the right side

The structure assistant uses a dictionary with around 3000 terms from CAD data construction process in Daimler AG. As shown in Fig. 16.12, based on the structure level of part, part number and the assigned feature set of the part, the first prediction list—*Sickenbild (bead layout)*, *Lastpfad (load path)*, *Cliploecher (clip holes)*, etc. is provided. Regardless of the prediction list, the names can also be filtered by entering the first letters of the word. For instance, when 'ves' is entered, then only the names start with 'ves' will be listed.

**Parameter assistant**: for different features, the third functional element, the parameter assistant, suggests the meaningful initialization values, based on the data from *the start-part information* and the *name of Feature Group*. For instance, for a feature *CYCS* (Absolute Coordinate System), the given input will designate:

*(Part Number), (Name of feature group).*

The suggested coordinate value will be given in the following form:

*(X, Y, Z).*

Together with the feature assistant and the structure assistant, the parameter assistant makes the contribution to simplify daily design work and to offer the possibility of semi-automating design work.

### *16.6.2 Best Practice*

*NeuroCAD* has learned the features from more than 21,000 CAD parts (with around 2.8 million features) and was widely rolled-out in thousands of workstations within Daimler AG. The current version (beginning of 2020) has reached 92% accuracy.[6] There will also be a mobile version in the future.

The development and deployment of *NeuroCAD* has been supported by Agile Software Development Methods [27] which will become the norm in continuous DEVS/OPS (Development & Operation) type of Engineering of the future following three essential paradigms:

- Individuality and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation responding to change over following a plan.

The *NeuroCAD* team has provided some best practices when implementing AI in industry:

***Think big, start small***. *NeuroCAD* has the vision to enable high-automated CAD design. Instead of starting with several functions in the beginning, they divide this big vision into small realistic problems and start with the one with high data availability. Implementation time of the first prototype takes only 2.5 months, with one person with 100% capacity and 3 persons with 20% capacity. Building quick prototypes will help earning the confidence from stakeholders in the early phase and thus it will very important to the success of the project.

***Involve stakeholders from the beginning onwards***. As already introduced in sub-chapter "*Standard process for Machine Learning projects*", a deep understanding of the existing problem and available data is an essential step in an AI project. In the kick-off phase, the *NeuroCAD* team organizes several workshops to communicate with key users and to deeply understand their potential challenges during the implementation phase. This ensures that the final digital product is delivered according to the actual business needs.

***In-house development***. Many companies tend to hire external consultants or developers and SW-coders to deploy new technologies. This will be difficult or at

[6] It is assessed based on the correctness of the top 3 recommendations from feature assistant.

least challenging for AI projects, since the first challenge they encounter will be the data access problem. Yet, today's companies have oftentimes not found an internal policy and security way to open their data repositories for outside SW-development companies. Besides, the lack of enterprise's own knowledge will also be a barrier during the implementation process. Therefore, the development team of *NeuroCAD* at Daimler all stems from inside the company, with a high degree of programming skills and knowledge of AI. The team was supported by *SCRUM* method and Speed Coach, one local team without bureaucratic organization, and constantly exchanged experiences with local AI experts within the company. This ensures not only the high development speed, but also the fully utilized existing enterprise knowledge. In other circumstances, however, especially in smaller and medium sized companies, where such critical in-house skills are not available or cannot be mobilized easily, outside help from research institutes and SW companies are necessary and useful.

## References


1. Norvig P, Russel S (2009) Artificial intelligence: a modern approach, 3rd edn. Prentice Hall, Upper Sadele River, NJ, USA
2. Legg S, Hutter M (2007) A collection of definitions of intelligence. Front Artif Intell Appl 157:17–24
3. Gardner H (1993) Frame of mind: theory of multiple intelligence. Basic Books, New York
4. Nakashima H (1999) AI as complex information processing. Mind Mach 9:57–80. https://doi.org/10.1023/A:1008322730047
5. Hopgood AA (2005) The state of artificial intelligence. In: Zelkowitz MV (ed) Advances in computers, vol 65. Academic Press, London, pp 1–75
6. Hopgood AA (2003) Artificial intelligence: hype or reality. Computer 36:24–28
7. Feigenbaum E, Buchanan BG, Lederberg J (1969) Heuristic DENDRAL: a program for generating explanatory hypotheses in organic chemistry. Mach Intell 4:209–254
8. Kohavi R, Provost F (1998) Glossary of terms. Machine learning—special issue on applications of machine learning and the knowledge discovery process. Mach Learn 30:271–274. https://doi.org/10.1023/A:1017181826899
9. Fayyad U, Piatetsky-Shapiro G, Smyth P Uthurusamy R (eds) (1996) Advances in knowledge discovery and data mining. American Association for Artificial Intelligence, Menlo Park
10. Mitchel TM (1997) Machine learning. MIT Press, McGraw-Hill Science, Cambridge
11. Géron A. (2017) Hands-on machine learning with Scikit-Learn and TensorFlow. O'Relly Media, Sebastopol
12. Sutton RS, Barto AG (2018) Reinforcement learning: an introduction. MIT Press, Cambridge
13. Ciaburro C (2018) Keras reinforcement learning projects: 9 projects exploring popular reinforcement learning techniques to build self-learning agents. Packt Publishing, Birmingham
14. Krüger J, Fleischer J, Franke J, Groche P (2019) WGP Standpunkt: KI in der Produktion. Künstliche Intelligenz erschliessen für Unter-nehmen. WGP Wissenschaftliche Gesellschaft für Produktionstechnik e.V., Hannover
15. Agrawal A, Gans J, Goldfarb A (2018) Prediction machines. Harvard Business Review Press, Boston
16. Chapman P, Clinton J, Kerber R, Khabaza T, Reinartz T, Shearer C, Wirth R (2000) CRISP-DM 1.0, CRISP-DM Consortium
17. Olson DL, Delen D (2008) Advanced data mining techniques. Springer, Berlin, Heidelberg
18. Stark J (2011) Product lifecycle management. In: Product lifecycle management. Decision engineering. Springer, London, pp 1–16. https://doi.org/10.1007/978-0-85729-546-0_1

19. Li J, Tao F, Cheng Y et al (2015) Big Data in product lifecycle management. Int J Adv Manuf Technol 81:667–684. https://doi.org/10.1007/s00170-015-7151-x
20. Jun H, Shin JH, Kim Y, Kiritsis D, Xirouchakis P (2009) A framework for RFID applications in product lifecycle management. Int J Comput Integr Manuf 22(7):595–615. https://doi.org/10.1080/09511920701501753
21. Abramovici M (2015) Smart products. In: The international academy for produ, Laperrière L, Reinhart G (eds) CIRP encyclopedia of production engineering. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-35950-7_16785-1
22. Ashton K (2009) That 'Internet of Things' thing, in the real world, things matter more than ideas. RFID J 22:97–114
23. Mukhopadhyay SC, Suryadevara NK (2014) Internet of things: challenges and opportunities. In: Mukhopadhyay S (ed) Internet of things. Smart sensors, measurement and instrumentation, vol 9. Springer, Cham. https://doi.org/10.1007/978-3-319-04223-7_1
24. Lee J, Bagheri B, Kao HA (2014) A cyber-physical systems architecture for industry 4.0-based manufacturing systems. Manuf Lett 3:18–23
25. Lee J, Ardakani HD, Yang S, Bagheri B (2015) Industrial big data analytics and cyber-physical systems for future maintenance & service innovation. Procedia CIRP 38:3–7. https://doi.org/10.1016/j.procir.2015.08.026
26. Yang S, Bagheri B, Kao HA, Lee J (2015) A unified framework and platform for designing of cloud-based machine health monitoring and manufacturing systems. J Manuf Sci Eng 4:137. https://doi.org/10.1115/1.4030669
27. Beck K, Grenning J, Martin RC, Beedle M, Highsmith J, Mellor S, Bennekum AV, Hunt A, Schwaber K, Cockburn A, Jeffries R, Sutherland J, Cunningham W, Kern J, Thomas D, Fowler M, Marick B (2001) Manifesto for agile software development. Available via Agile Alliance. https://agilemanifesto.org. Accessed 6 Nov 2020