Aalto University
School of Chemical
Engineering

James Clovis Kabugo

MONITORING THE WASTE TO ENERGY PLANT USING THE LATEST AI METHODS AND TOOLS

Master´s Programme in Chemical, Biochemical and Materials Engineering
Major in Chemical Engineering

Master's thesis for the degree of Master of Science in Technology submitted for inspection, Espoo, 26 January 2018.

Supervisor Professor Sirkka-Liisa Jämsä-Jounela

Instructor Dr. Bei Sun

Aalto University
School of Chemical
Engineering

Aalto University, P.O. BOX 11000, 00076 AALTO
www.aalto.fi

Abstract of master's thesis

Author James Clovis Kabugo

Title of thesis Monitoring the waste to energy plant using the latest AI methods and tools

Department Chemical and metallurgical Engineering

Major Chemical Engineering

Thesis supervisor Professor Sirkka-Liisa Jämsä-Jounela

Thesis advisor(s) / Thesis examiner(s) Dr. Bei Sun

Date 26.01.2018 **Number of pages** 84+2 **Language** English

Abstract

Solid wastes for instance, municipal and industrial wastes present great environmental concerns and challenges all over the world. This has led to development of innovative waste-to-energy process technologies capable of handling different waste materials in a more sustainable and energy efficient manner. However, like in many other complex industrial process operations, waste-to-energy plants would require sophisticated process monitoring systems in order to realize very high overall plant efficiencies. Conventional data-driven statistical methods which include principal component analysis, partial least squares, multivariable linear regression and so forth, are normally applied in process monitoring. But recently, latest artificial intelligence (AI) methods in particular deep learning algorithms have demostrated remarkable performances in several important areas such as machine vision, natural language processing and pattern recognition. The new AI algorithms have gained increasing attention from the process industrial applications for instance in areas such as predictive product quality control and machine health monitoring. Moreover, the availability of big-data processing tools and cloud computing technologies further support the use of deep learning based algorithms for process monitoring.

In this work, a process monitoring scheme based on the state-of-the-art artificial intelligence methods and cloud computing platforms is proposed for a waste-to-energy industrial use case. The monitoring scheme supports use of latest AI methods, laveraging big-data processing tools and taking advantage of available cloud computing platforms. Deep learning algorithms are able to describe non-linear, dynamic and high demensionality systems better than most conventional data-based process monitoring methods. Moreover, deep learning based methods are best suited for big-data analytics unlike traditional statistical machine learning methods which are less efficient.

Furthermore, the proposed monitoring scheme emphasizes real-time process monitoring in addition to offline data analysis. To achieve this the monitoring scheme proposes use of big-data analytics software frameworks and tools such as Microsoft Azure stream analytics, Apache storm, Apache Spark, Hadoop and many others. The availability of open source in addition to proprietary cloud computing platforms, AI and big-data software tools, all support the realization of the proposed monitoring scheme.

Keywords: Monitoring, Waste to-energy, Artificial intelligence, Deep learning, Cloud computing

Acknowledgements

First, I would like to thank my supervisor Professor Sirkka-Liisa Jämsä-Jounela for offering me the opportunity to work on this interesting research topic and for the support and guidance provided in completion of this work. Also I take this chance to thank my instructor, Dr. Bei Sun for his effort and technical support. Furthermore, I would like to say thank you, to all my colleagues in the laboratory of process control and automation at Aalto.

Finally, I extend my special thanks to Outotec (Germany) for all their support of this work.

Espoo, 26.01.2018

James Clovis Kabugo

Contents

Nomenclature

Latin alphabet

A	constant	–
a	constant	–
B	constant	–
C	constant	–
c_p	specific heat capacity at constant pressure	J/(kgK)
D	constant	–
T	temperature	K
m	mass flow	kg/s
p	pressure	Pa
Q	heating value (enthalpy)	J/kg
q	valve opening	%
V	volume flow of air	Nm^3/h
X	matrix	–
x	variable	–
Y	vector	–
y	variable	–

Greek alphabet

β	vector of model coefficients
Δ	observed change between variable(s)
ε	noise or errors
λ	box-cox transformation parameter

Subscripts

n	column number
p	row number
t	time

Abbreviations

AE	auto-encoder
AI	artificial intelligence
AMQP	advanced message queuing protocol
ANOVA	one-way analysis of variance
APCA	adaptive principal component analysis
APIs	application programming interfaces
AWS	amazon web services
CNNs	convolutional neural networks
CoAP	constrained application protocol
CPU	central processing unit
CUDA	compute unified device architecture
CVA	conical variate analysis
DBM	deep Boltzmann machine
DBN	deep belief network

DNNs	deep neural networks
FDA	Fisher discriminant analysis
GPU	graphics processing unit
HTTP	hypertext transfer protocol
IoT	internet of things
I/O	input/output
IT	information technology
JDBC	java database connectivity
KICA	kernel independent component analysis
KVM	kernel-based virtual machine
LSTM	long short time memory
M2M	machine-to-machine
MQTT	message queue telemetry transport
MWPCA	multiway principal component analysis
NPE	neighborhood preserving embedding
ODBC	open database connectivity
OLEDB	object linking and embedding database
OPC	OLE (object linking and embedding) for process control
OPC-UA	OPC unified architecture
O/S	operating system
RBM	restricted Boltzmann machine
REST	representational state transfer
RNNs	recurrent neural networks
SDK	software development kit
VIP	variable importance in projection
VS	visual studio

List of Figures

List of Tables

1 Introduction

1.1 Historical Background

The origins of artificial intelligence (AI) can be traced back in the 1950s following Turing's paper titled, "Computing Machinery and Intelligence". [1]. Alan Turing in his paper proposed 'the imitation game' as one way to check if machines can be able to think comparably to humans. This was later referred to as the 'Turing test' and it has been used as a measure of artificial intelligence progress over the years. [2]. However, John McCarthy [3-5] is credited for the term 'artificial intelligence', back in 1956. In the 1950s, research in AI started to grow and lead to several successes. During this time AI was utilized in solving problems which were considered difficult for people but rather simple for computers to handle [6] using formal mathematical rules. For example, in 1957 Frank Rosenblatt [3, 7] developed the perceptron, the basic artificial neural network that can be applied in pattern recognition. The perceptron is considered a remarkable step towards artificial intelligence. At the end of 1950s, the term 'machine learning' emerged, invented by Arthur Samuel from IBM. [3] The idea was to programme computers so that they can learn and solve problems for instance playing games, checkers or chess even better than humans. Later, in the early sixties the first industrial robot called 'Unimate' [3] was realized at General Motors plant, in the US. Also notable, in the mid-sixties researchers at Stanford University developed what is considered as the first 'expert system', DENDRAL, which was applied in decision-making and analysing the chemical structure of organic compounds. This paved way for many other early expert systems such as XCON (eXpert CONfigurer) for computer hardware configuration, MYCIN used in bacterial infection diagnosis and ACE for cable maintenance. Companies involved in the development of expert systems included IBM, HP, and Xerox, among others. [3-5]

Although progress of AI was noticeable, it slowed down in the 1970s in what is known to many in the field as the 'AI winter'. James Lighthill in his report [3-5, 8] to the British government showed that, based on the results from AI projects at the time, AI had failed to meet its intended promises and therefore not worthy of further research support. The promises were based on performance indicators in advanced automation, research concerning the central nervous system and robotics. The report showed AI progress in automation and study of the central nervous system was limited and building robots was not worthwhile. This lead to significant reduction in AI funding by the British government. [4] A similar effect was felt in the US, where the ultimate goal of AI was to succeed in machine translation. In that case, the Automatic Language Processing Advisory Committee (ALPAC) report implied that the idea of machine translation was 'hopeless' [2] and many saw AI as a disappointment [3]. The US government proceeded by heavily slashing AI funding. [2-5]

In the early 1980s AI growth picked up again, for instance cognitive researchers developed models based on symbolic reasoning and later cognitive models were implemented in neural networks models. The idea of 'connectionism or parallel distributed processing' [6] was developed during this time, a key factor in AI development. In the same period, AI was spurred by the ambitious Japanese project of building a fifth-generation computer. The Japanese project was designed with a goal of creating a 'parallel-inference machine' with listening and speaking capabilities. [3, 9] However, despite the huge financial investment (equivalent to $850 million) injected in by the Japanese government, this venture did not yield the intended outcomes. A result which raised cautiousness within

the AI community, similarly to setbacks in the 'AI winter' in the seventies [3-5]. Also a notable AI project in the 1980s, is the Cyc project at Stanford University. In this case researchers embarked on a task of manually compiling data and knowledge based on ontology and human common sense knowledge into one collection named the 'encyclopaedia of knowledge (Cyc)'. [6, 9] This project is still in progress and maintained by Cycorp Inc. But its development declined due to the emergence of internet search engines and current availability of big data [9]. Despite the fewer successes coupled with setbacks, AI development continued to thrive even though at a relatively slow pace. There were several lessons learnt from the successes and failures. For example, following the unsuccessful Japanese project, researchers started to focus more on developing AI software and only develop hardware to support the performance of the former. [9] In the Cyc case, the idea of exporting huge amounts of human knowledge to machines proved cumbersome and complex. Problems associated with AI systems designed based on 'hard-coded knowledge' lead to new ideas such as the need for AI systems to acquire knowledge on their own, for example through feature extraction from raw data. [6, 9]

Although AI growth was slow through the 'AI winters', the research continued and in 1997 a milestone was reached when IBM's 'Deep Blue' supercomputer became the first computer to defeat the reigning world chess champion, Gary Kasparov. [4] But a big accomplishment happened in 2006, when Hinton et al. [6] published the architecture of deep belief neural networks and how they can be trained. This led to a boom in AI research and investment. Several AI projects and applications software, databases sprung up. Examples include ImageNet (image database), Google's app for speech recognition, IBM's Watson (cognitive business platform), Google DeepMind's AlphaGO (a GO-playing program) and Baidu's Duer (a personal assistant). In fact, the Turing's test was passed in 2014 after over 60 years by a chatbot named Eugene Goostman. [4] A milestone was reached and an indication that AI has developed and probably will achieve most of its intended promises later on. The application of deep neural networks (DNNs) also called 'Deep Learning' has fuelled the resurgence of artificial intelligence. [6]

1.2 Motivation

The impetus behind this work is the fact that currently the AI field is rapidly growing, opening new opportunities and believed to be the future norm in many real-life applications. Looking at the development of artificial intelligence, it can be described in three waves, which include handcrafted knowledge, statistical learning and contextual adaptation. The first wave represents AI systems involving human-engineered rules mainly based on reasoning, without learning capabilities and poor in handling uncertainties. Expert systems, inference systems are among the AI systems in that category. The second wave was dominated with development of statistical models and training such models on big data. These AI systems exhibit perception and learning capabilities in addition to considerable reasoning ability. Machine learning methods and notably deep learning are key to the resurgence of artificial intelligence. The new and third wave of AI can be said to have started in 2010. It aims at building systems with attributes like perception, learning, knowledge abstraction and reasoning. This current AI wave is considered as the intersection of the two previous waves. [10-11]

The remarkable performance of deep learning algorithms and big data availability have led to the rapid growth of current AI research in a wide range of applications. The successes of deep learning

has energized both researchers and investors into AI research and development to build intelligent systems for specific industrial applications such as in retail business, medical diagnosis, military, transportation and industrial automation. This current wave of AI has led to increasing number of AI start-ups and big companies such as google, Apple, Amazon and Baidu have acquired and merged several of these startups. For example, google acquired a small neural network company called 'DNNresearch' [9] at a considerably high price and Intel acquired a smart vision company, Mobileye for $15 billion [12]. Current applications of AI technologies include image and speech recognition, pattern recognition, natural language translation, computer vision, robotics, games and so forth.

Moreover, according to the International Data Corporation (IDC) prediction [13], revenues from cognitive and artificial intelligence systems will reach $12.5 billion this year, showing an increase of 59.3% when compared to 2016. Furthermore, the report' forecast for 2020, show that the revenues will be over $46 billion based on the observed increase in global spending on cognitive and AI technologies. [13]

Due to the increased interest in this area of research and development, more and more industries and even governments [14] are interested in integration of the AI in their routine operations. Notable leading countries in AI development include US, Japan, China and several countries in Western Europe. Most popular technology companies are constantly developing AI systems targeting various business sectors. For example, IBM developed Watson an intelligent cognitive system [15] which has been applied in the health sector to assist in decision-making during clinical diagnosis.

Furthermore, there is a growing consensus that the fourth industrial revolution (Industry 4.0) is underway, powered by smart automation or more broadly by artificial intelligence. It is anticipated that AI will lead to new innovations, improved productivity and creation of opportunities. Like with other industries, the process industry will have to adjust with artificial intelligence development for it to be more productive, profitable and sustainable. [16]

1.3 Scope

This work emphasizes the use of latest of AI methods and software tools in process industry and examines their application in process monitoring. Currently, in the artificial intelligence field, deep learning is a hot topic and a major contributor to AI resurgence. Therefore, this work explores available deep learning software tools both commercial and open source tools. Application of artificial intelligence, specifically deep learning algorithms in industrial processes is briefly studied. Other areas of AI such as different types of machine learning methods and related areas like cognitive computing, big-data and cloud computing are also briefly examined. Furthermore, the work applies a waste-to-energy industrial case study in development of a process monitoring system based on the current wave of AI.

1.4 Aims

The major objective of this study was to develop a process monitoring system which employs the state-of-the-art artificial intelligence methods, big data processing tools and cloud computing platforms. Furthermore, the work aimed at exploring how such a monitoring system can be implemented in a waste-to-energy industrial case study to achieve real-time monitoring of process operations.

Literature Review Part

2 State-of-the-art artificial intelligence methods and tools

Artificial intelligence (AI) can be defined in a general sense as the ability of machines to demonstrate intelligence in away comparable to humans. [17-19] Artificial intelligence is a computer science field which deals with development of software systems to enable machines or devices to learn and behave like humans. The goals of artificial intelligence include knowledge representation, planning, natural language processing, learning, perception, reasoning, problem solving, among others. The AI field is quite broad, there are many approaches and tools that can be used to achieve different AI goals. Approaches to AI include cognitive simulation, logic, knowledge-based, machine learning and statistics, and so forth. Examples of AI tools in this context include 'search and optimization', artificial neural networks, logic, control theory, programing languages and statistical learning methods. [20-21]

The evolution of artificial intelligence has been categorized into three main stages: artificial narrow intelligence (ANI), artificial general intelligence (AGI) and artificial superintelligence (ASI). The first stage, ANI, is also known as 'weak AI' partly because it has a limited scope of intelligence. It specializes only in 'one functional area'. [22-23] In fact this represents the level of AI developed in the past 60 years. The second stage, AGI, can handle multiple functional areas such as reasoning capabilities, problem solving and abstract thinking. Its intellectual capabilities are comparable to human intelligence. It can be said that AGI has not been realized, however, Figure 1 suggests that we are much closer than ever before. Transition from ANI to AGI is predicted to occur by 2020. The emergence of new technologies, social demands and data resources are accelerating the transition from ANI to AGI. The last stage, ASI, represents the level of AI where machines are smarter than humans in all areas. According to Figure 1, this is predicted to happen in over 30 years from now provided AI development remains at the current growth rate. [23]

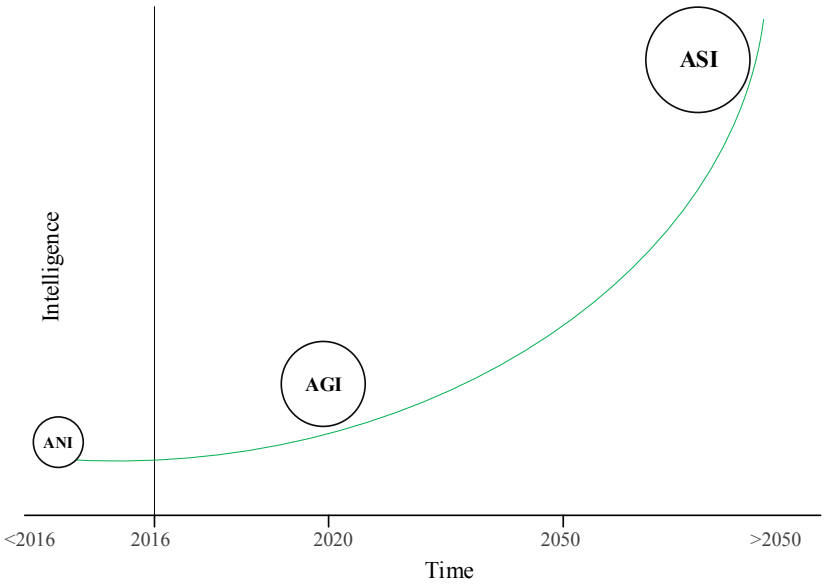

Figure 1. The three stages of artificial intelligence. [23]

2.1 Artificial intelligence regional outlook

At the moment North America and specifically the US is leading in many aspects of artificial intelligence development such as start-ups, government funding and number of innovative AI companies. The US also has many of the world leading technology firms like IBM, Microsoft, Google, Apple, and Facebook which are heavily investing in AI technologies. The Asia Pacific region is growing rapidly with China as the major key player alongside Japan. China is among the leading countries in AI research based on paper publications. Artificial intelligence research in China is supported largely by the private sector. Availability of large volumes of data in addition to high computing capacity is expected to spur rapid AI growth in the Asia Pacific region. Western Europe has also seen increased growth in artificial intelligence with Britain and Germany leading the way. Figure 2 shows funding and the number of AI companies in selected countries in 2016. [24-27]

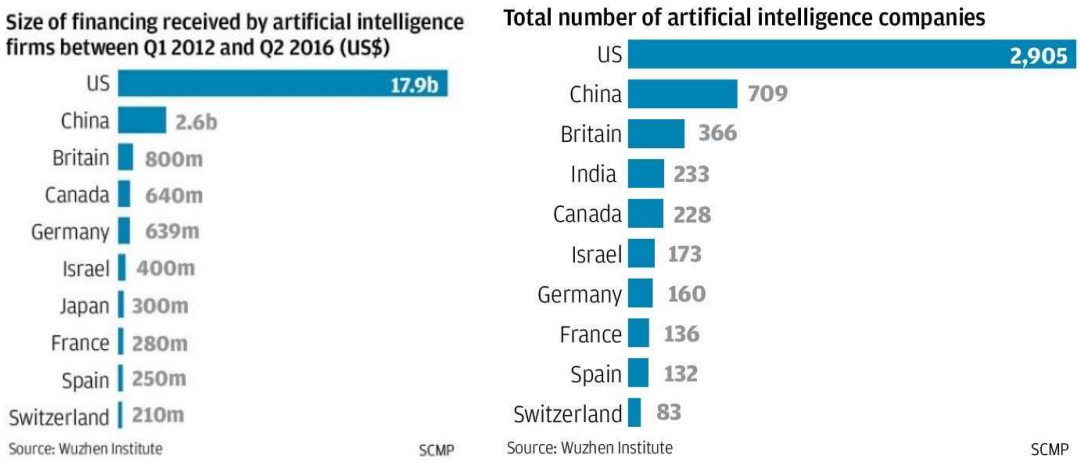

Figure 2. Some of the key AI countries: funding and number of AI companies. [27]

2.2 Big data and artificial intelligence

Big data can be described as massive volumes of data sets obtained from several sources such as data collected by sensor networks, social media, mobile devices, electronic or digital transactions, administrative centres, financial institutions, industrial operations, scientific research centres and so forth. [28-31] Data can be in a structured, semi-structured or unstructured form. Structured data is well categorized or labelled data. However, most of the data available falls under the unstructured category which is difficult to process into useful information. Examples of unstructured data types include, videos, audio, digital images, posts on social media, among others. Big data also can be defined based on the 'four Vs' which include volume, velocity, variety and veracity. In some sources, big data is defined by more than 'four Vs' as shown in Figure 3. [30]

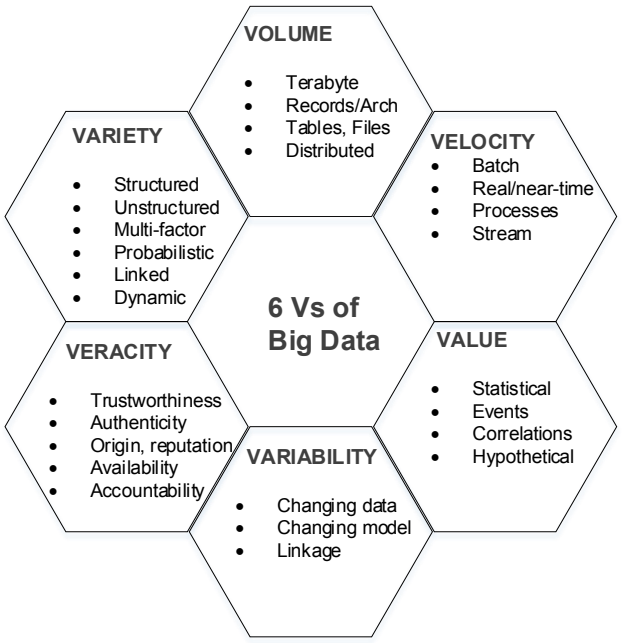

Figure 3. The 6 V's of big-data. [30]

In regard to the 'four Vs' of big data, volume describes the massiveness of data sets and the requirement of larger storage capacities than in traditional databases. Moreover, a variety of data is collected for example in form of videos, images, text, and audio, among others. As shown in Figure 3 and mentioned earlier, the data can be structured or unstructured with the latter being more problematic to handle and utilize. Big data is also characterized by velocity or rate at which data is collected. In some cases, huge volumes of data are continuously collected for example online streaming. Furthermore, in many cases large volumes of data are collected at high rates. Storing and processing real-time big data also presents quite significant challenges. It is also important to mention that sometimes data obtained may be unauthentic, unreliable or not useful at all. Therefore, data filtering is necessary but handling big-data is a challenge. [29-30]

Big data is creating several opportunities in different sectors such as retail, communication, health care, cyber security, financial and in industrial automation, among others. However, the 'explosion' in the amount of data generated also presents difficulties especially on how to harness this kind of data into useful information. The complexity of big data due to numerous data forms and often with very high dimensionality, adds to the challenges in its processing. For instance, some data maybe obstructed with noise or errors and thus unreliable without thorough pre-processing. This implies that sophisticated data processing methods are required to leverage big data. [28, 30-31]

Artificial intelligence methodologies are being used to process big data into useful information which can be utilized in decision-making. In fact, the availability of big data has contributed greatly to an overwhelming interest in artificial intelligence in recent years. Development of machine learning algorithms and more specifically deep learning to harness big data is among the hottest topics today. However, training deep learning algorithms on considerably large data sets is computationally demanding. [32] Nevertheless, powerful computer hardware and software platforms which support distributed and parallel computing are increasingly becoming available from numerous high technology manufacturers such as IBM, Oracle, Microsoft, Intel and NVIDIA, among others. Convergence of big data and AI is expected to lead to new innovations, improved services and better decision-making in different areas including health care, business, manufacturing, environment sustainability and so forth. [28-32]

2.3 Cognitive computing and artificial intelligence

Cognitive computing is a technological approach which enables human-machine collaboration [33-36]. It involves application of artificial intelligent methods in order to understand the human mind. Cognitive systems are considered as 'self-learning systems' that apply machine learning, reasoning, vision and natural language processing techniques among others, to simulate the functions of the human brain. Cognitive computing is still in the early stages of development but together with AI technologies, are expected to transform the processing of big data into useful knowledge. Thus leading to key innovations in several areas of real-life applications. [33] The fundamental principles of cognitive systems include ability to learn, model development and the use of a probabilistic approach. Cognitive systems also display attributes of being adaptive, iterative, contextual and stateful as well as being interactive. Among the notable AI technologies with cognitive abilities include IBM's Watson, Google's Deepmind, Numenta, SparkCognition, Microsoft cognitive services and Expert System. [34] Figure 4 demonstrates the role of cognitive systems in data processing according to IBM's Watson a known cognitive platform.

In the process industry, there is a growing interest in cognitive systems in areas like product quality analytics, process control, process monitoring and automation. For instance, a commercially available cognitive system, 'IBM Watson IoT Platform' can be used in detection of potential equipment failures within the process more efficiently than conventional process control methods. It can be applied in product quality monitoring, calibration of process instruments and so forth. A combination of cognitive computing and AI offers transition from systems governed by 'rigid rule-based algorithms' to more flexible, efficient and autonomous systems. [33-36]

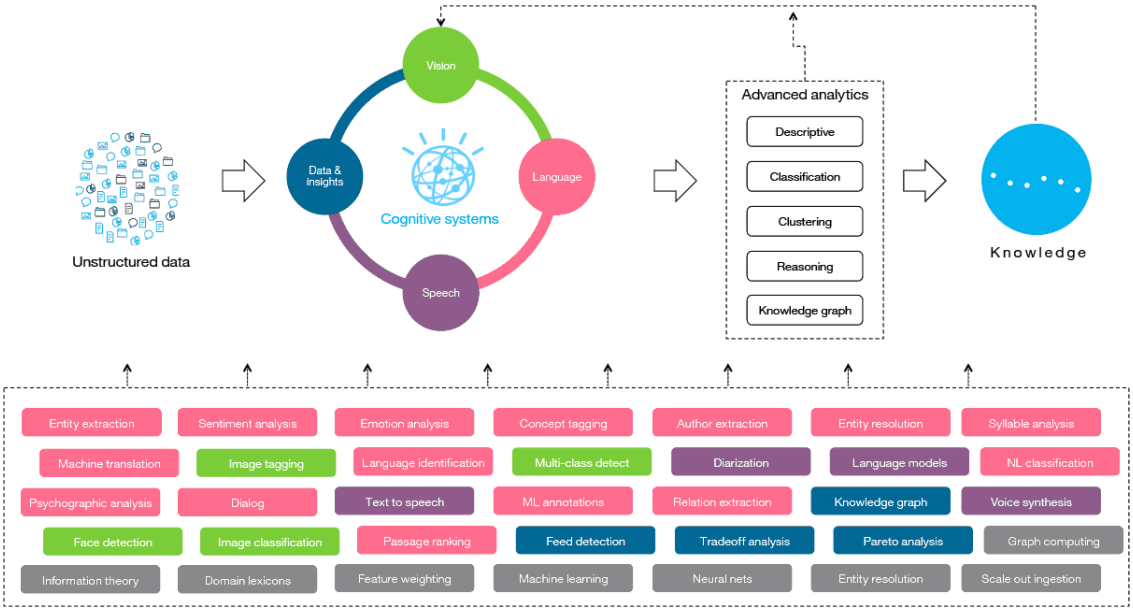

Figure 4 Schematic representation of data processing using cognitive systems. [36]

2.4 Machine learning and artificial intelligence

Machine learning is a field in computer science dealing with statistical modeling and provides key approaches to artificial intelligence. It is concerned with programming computers to learn from past experience or from examples of data sets. After training a machine with a set of data, it is expected to execute similar tasks on unfamiliar data sets. Machine learning can be divided into two major categories, supervised learning and unsupervised learning as described in Figure 5. But sometimes there combinations of these two strategies known as semi-supervised learning. Another special machine learning method is reinforcement learning. This is a special AI method because it overlaps other artificial intelligence approaches in addition to machine learning. [37-39]

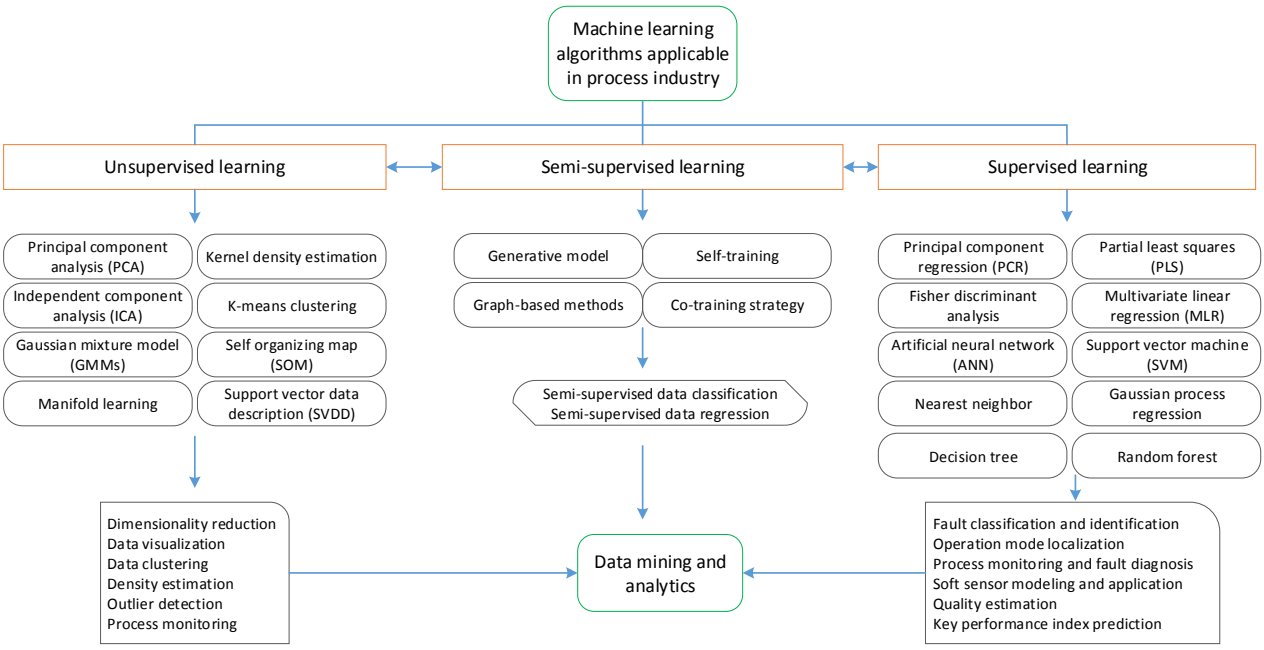

Figure 5. Machine learning methods. [40]

2.4.1 Supervised learning

In supervised learning the target is to learn a mapping from inputs to outputs for a given input-output data set. Here the 'supervisor' provides a training dataset. This type of machine learning is also referred to as 'predictive' learning technique. It is mainly applicable in classification or pattern recognition problems as well as in regression as shown in Figure 5. [36-39]

2.4.2 Unsupervised learning

In unsupervised or descriptive learning, only the output data is available. The objective is to find the 'interesting structure' within the data set, which is sometimes referred to as 'knowledge discovery'. Since the input data is not available, the computer must estimate the solutions to the given data set on its own. In other words, the idea is to find 'what generally happens and what does not', statistically known as 'density estimation'. As previously shown in Figure 5, unsupervised machine learning includes clustering and dimension reduction algorithms. [36-39]

2.4.3 Machine learning software tools

There several machine learning (ML) software tools available both commercial and open source software. Popular commercial machine learning software is provided as machine learning as a service (MLaaS). Examples of the leading proprietary MLaaS include Amazon machine learning (Amazon ML), Microsoft Azure machine learning, Google machine learning and IBM Watson machine learning. These machine learning tools are accessible through respective cloud platforms. They offer highly scalable environments and a variety of machine learning algorithms for predictive analysis among other functions such as data pre-processing. Moreover, most of the commercial ML frameworks also provide deep learning libraries and big-data computing software tools like Apache Spark, Hadoop, etc.

In the case of open source ML software tools, a number of machine learning libraries, frameworks are freely accessible and can also be implemented either via the cloud or on-premises. Notable examples of machine learning software tools are Apache Spark MLlib, Scikit-learn, TensorFlow, H2O.ai, Big ML, Accord.NET, Apache SystemML, Apache Mahout, Oryx 2 and so forth. Moreover, some of the existing open source machine learning libraries also have been incorporated into proprietary ML platforms. For instance both Apache Spark MLlib and H2O.ai are available in Microsoft Azure HDInsight, a big-data analytics cloud service platform. This implies that a number of machine learning algorithms from different software libraries can be accessed under a single machine learning platform. Figure 6 illustrates how machine learning algorithms in Microsoft azure ML Studio [41] can be utilized in building data-based predictive models.

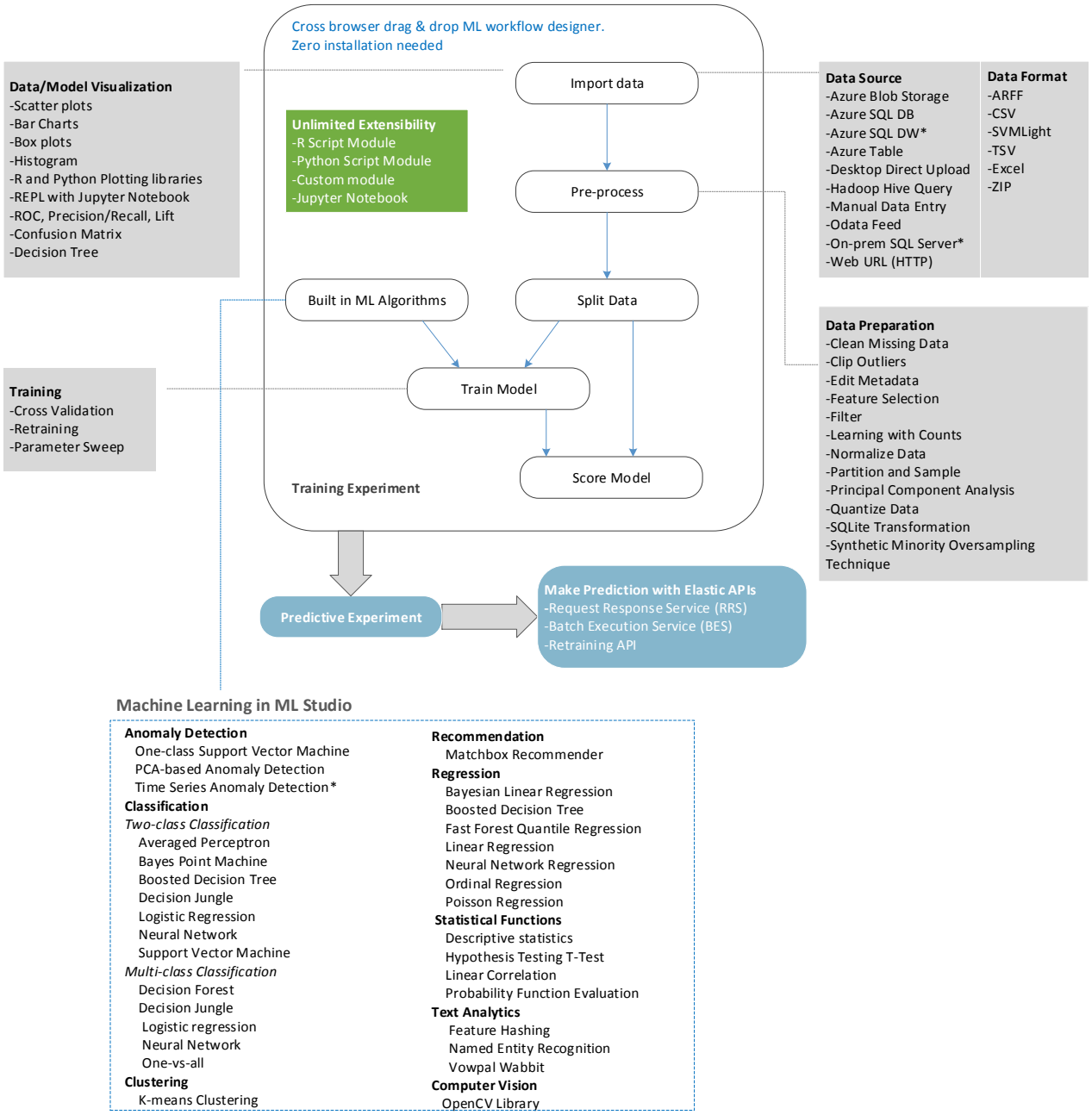

Figure 6. A scheme showing Microsoft Azure ML Studio capabilities. [41]

2.5 Reinforcement learning

Reinforcement learning can be considered a machine learning technique in which learning is a trial and error approach when dealing with sequential decision-making tasks. [36, 42-45] The decisions made at any stage or state are influenced by rewards and penalties. An example of reinforcement learning in practice, is how a child learns to walk [36]. Also there several decisions which humans have to make in order to navigate their way through life in a complex and uncertain world environment. Therefore, it can be said that the aim of reinforcement learning is to find optimal decisions for a particular problem. The strategy that leads to an optimal decision corresponds to the maximum and long-term 'reward' [45]. In other words reinforcement learning algorithms try optimize the sequential decision making actions when an 'agent' (e.g. robot or computer) is interacting with the

environment over time. Reinforcement learning in many aspects falls under unsupervised learning category of machine learning methods. Application of reinforcement learning has been limited to systems where key features are extracted manually and on a low dimensional data. However, in deep reinforcement learning, agents are able to learn and build their own knowledge [45] directly from raw input data without use of hand crafted features characteristic of supervised learning. The idea of reinforcement learning is demonstrated in Figure 7.

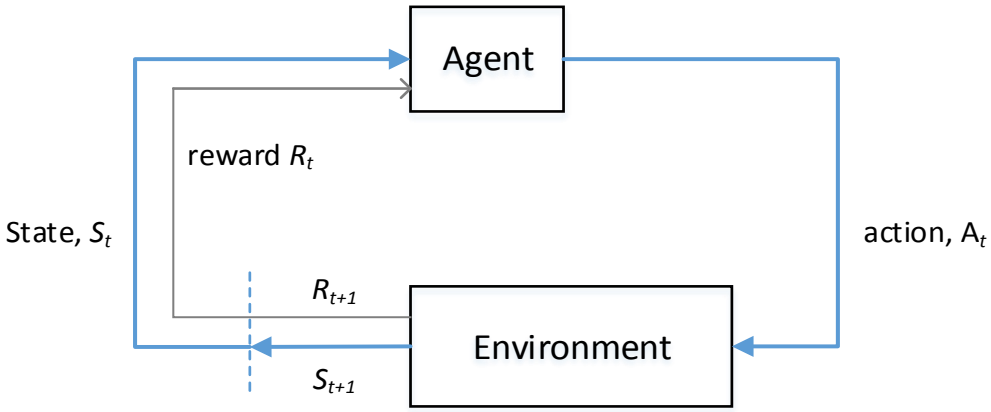

Figure 7. Schematic representation of reinforcement learning. [37]

2.6 Deep learning and artificial intelligence

Deep learning is one of the many different machine learning techniques. It is a type of representation learning approach within machine learning. The essence of deep learning is to learn the hierarchal representations of data sets. Although deep learning technique has been known for decades, it only became popular since 2006. [6] Currently it's one of the major drivers of the 'third wave' of artificial intelligence. In many machine learning fields such as object recognition, speech recognition, image segmentation and machine translation, deep learning is taking the leading role after performing better than the respective traditional methods. According to LeCun et al. [46], deep learning can be defined as a machine learning technique which permits 'computational models that are composed of multiple processing layers to learn representations of data with multiple levels of abstraction'. Deep learning algorithms are able to extract and learn intricate structures from highly dimensional data sets automatically instead of hand-crafted feature extraction approach [6, 46-47]. In deep learning the key representations or features from data sets are learned automatically using a 'general-purpose learning procedure'. [46] Deep learning enables machines to learn from data and improve with experience. In recent years, deep learning algorithms have been successful in solving certain problems like playing the 'Go game', which had eluded other artificial intelligence approaches. In this work, only a shallow discussion about deep learning is presented with the emphasis put on its application in process industry related problems. The concept of deep learning was developed based on artificial neural networks. It is currently implemented through variants of multilayer neural networks such as multilayer perceptron. However, probably in future, the concept can be applied to other machine learning approaches such as decision trees. Figure 8 illustrates the general concept of deep learning in artificial neural networks. [48]

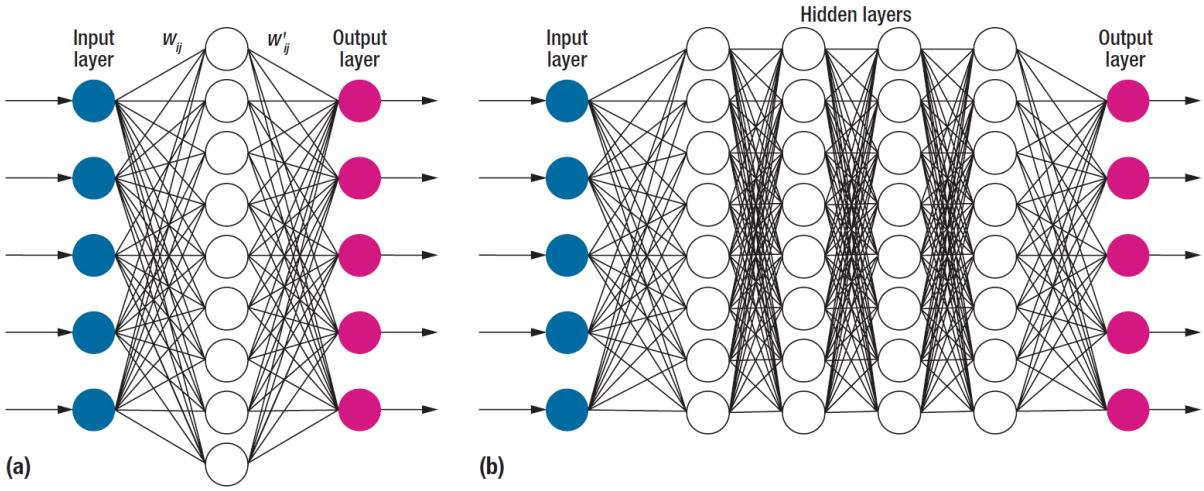

Figure 8. Artificial neural networks: (a) shallow neural network and (b) deep neural network. [48]

2.6.1 Deep learning architectures

In general, deep learning architectures can be categorized into three classes: unsupervised (or generative), supervised (or discriminative) and 'hybrid' deep architectures. [49] This categorization is similar to that discussed earlier in machine learning methods. There are several deep learning models falling under these three categories mentioned above. Details of these are not discussed here, only few examples are given. Among the commonly applied deep learning models include deep auto-encoders, restricted Boltzmann machine (RBM), convolutional neural networks (CNNs) and recurrent neural network (RNNs).

Auto-encoders (AE) neural networks consists of an encoder and a decoder. The purpose is to learn the representations of the input data (encoding) and be able to generate the output that is similar to the original input data (decoding). This means that the dimensions of the input and output are equal. It is a special kind of a deep neural network (DNN) under unsupervised deep learning algorithms. There are several variants of auto-encoders such as sparse, denoising and stacked auto-encoders [49]. Figure 9 illustrates the general structure of an auto-encoder model.

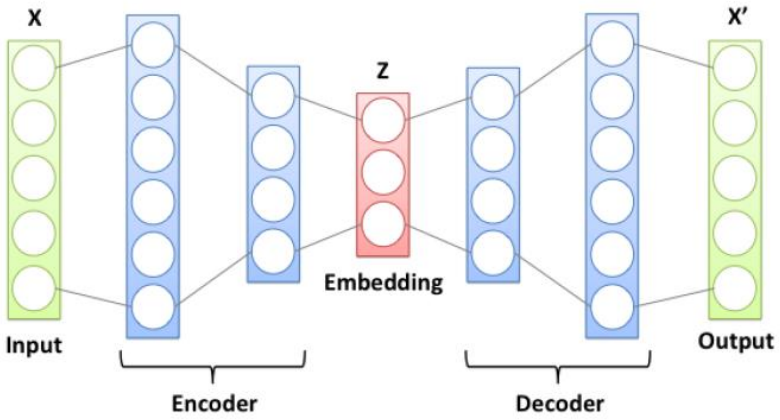

Figure 9. A general architecture of an auto-encoder model. [50]

The other example of deep learning architecture is the deep model of restricted Boltzmann machine (RBM). A restricted Boltzmann machine model is a special network restricted to a form of a 'bipartite graph'. When layers of restricted Boltzmann machine are stacked together, a deep belief networks (DBN) or a deep Boltzmann machine model is created as shown in Figure 10. A DBN is a graphical model with a mix of directed and undirected networks. In DBM, the model is a completely undirected network graph. Another difference between DBN and DBM is that, in the former, training is done layer-wise whereas the latter is trained as a connected model. [51]

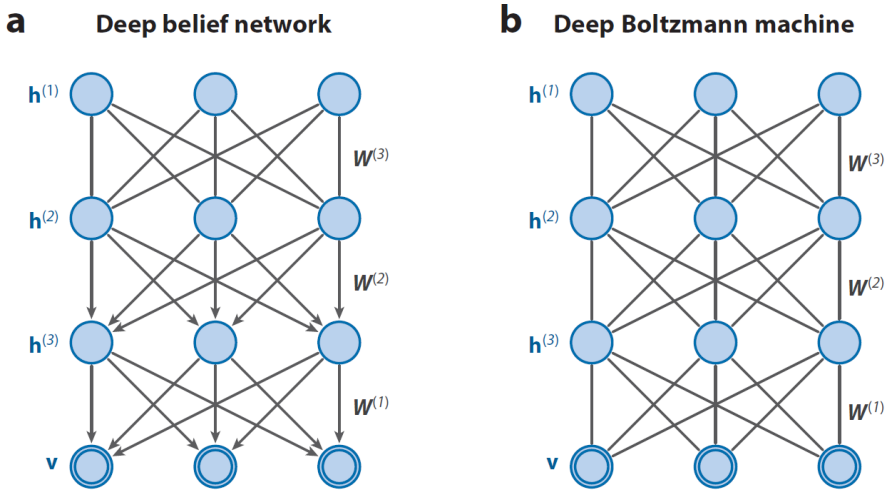

Figure 10. Deep models of restricted Boltzmann machine: (a) Deep belief and (b) Deep Boltzmann machine, where *v* is the input layer, *h* is hidden layer and *w* is weights. [51]

Convolutional neural networks (CNNs) are widely applied in the various fields such as image processing, natural language and speech recognition. In this case, the architecture is suitable for analyzing 2-D input data sets like images and speech signals. One of the key characteristics of the deep CNNs is the inclusion of multiple convolutional layers and dense (fully connected) layers within the network architecture. In addition to convolutional layers and dense layers, pooling (down sampling) layers are normally included in the CNN architecture with the aim of reducing the number of key features to be extracted from input data thereby minimizing the number of model parameters. Training of CNNs is much easier than that of fully connected networks. Figure 11 demonstrates the layers of convolutional neural network. [52-53]

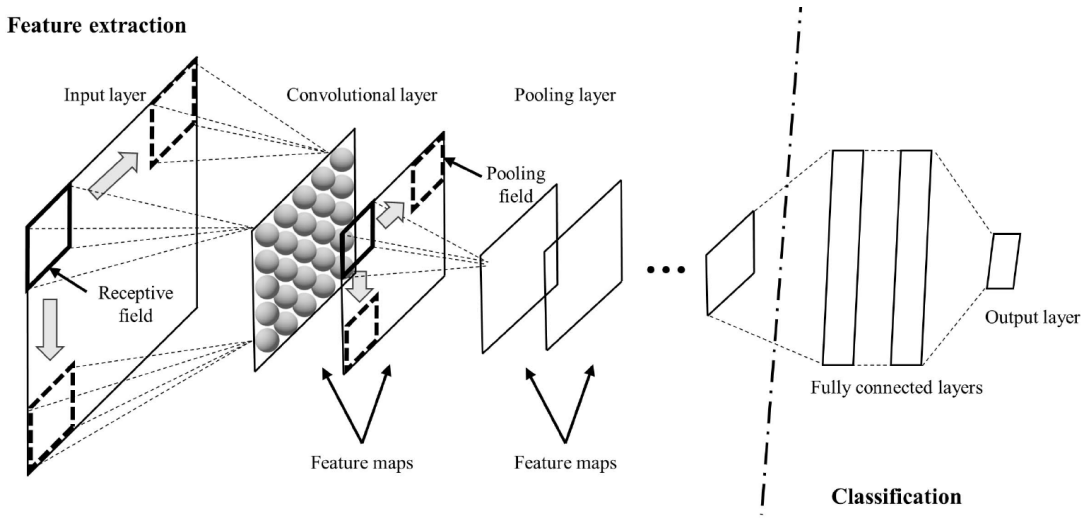

Figure 11. A simplified structure of a convolutional neural network. [54]

Recurrent neural networks (RNNs) are another type of deep learning algorithms. They are considered the 'deepest' among all the deep neural networks. The RNNs are applied in solving variable-sequence problems. In feedforward deep neural networks the idea is to solve the task in a hierarchal manner i.e. each layer partially computes the task before passing it to the next layer until the final output is generated after the last layer. However, RNNs differ from this approach by introducing the aspect of memory storage in between layers leading to a more complex network structure. An example is the Long-Short Time Memory (LSTM) network models. [55-56]

2.6.2 Deep learning algorithms in process control and monitoring

In literature, most of the studies concerning use of artificial intelligence in process control and monitoring refer to use of traditional machine learning algorithms (Figure 5) as well as other AI approaches like expert systems. However, there are several research works in which deep learning algorithms have been utilized in handling large process data. Here only a few examples of the previous studies related to application of deep learning algorithms in process control and monitoring are briefly discussed.

For instance, recently Deepa and Baranilingesan [57] proposed a deep learning based model predictive control (MPC) method for a continuous stirred tank reactor (CSTR) considering non-linearity behaviour. Deep learning algorithms were used in training data, in other words for predicting the plant model and the weights of the deep learning model were tuned by a method based on Particle Swarm Optimization (PSO) and Gravitational Search Algorithms (GSA) methods. The performance of the deep learning based MPC was compared to that of conventional controllers which included proportional-integral (PI) and proportional-integral-derivative (PID) controllers. The authors observed better performance of the deep learning algorithm tuned by a 'hybrid' PSO-GSA method in terms of obtaining a minimum integral square error as compared with the traditional controllers. Consequently, the deep learning based MPC was more effective in controlling the CSTR reactor than the conventional controllers. In a similar work, Lenz et al. [58] also studied controlling real-

time complex process dynamic behaviour with use of a 'deep MPC' model. The authors demonstrated their work on robotic food-cutting machine and cutting a wide range of food materials. The idea was to have the machine decide which tools to use based on the food properties at hand. It was perceived that, developing controllers for each individual food item would be cumbersome. Moreover, obtaining representative hand-crafted predictive models in each case would be very difficult or infeasible. After implementing deep learning combined with the MPC, the cutting rate and accuracy among various food items significantly improved. [58]

Spielberg et al. [59] reported a process control approach based on deep reinforcement learning to address some of the pitfalls of traditional control strategies such as complex dynamics and high dimensionality, among others. Here, the controller represented the agent and the process being the environment according to Figure 7. The approach uses deep neural networks to learn the control policies. After training, a policy that represents the process output to controller actions is then obtained. The method eliminates the need of hand-engineered features, controller tuning, derivation of control laws and creating mathematical models depicting process behaviour.

Deep learning algorithms have been applied in 'data-driven' machine health monitoring. Zhao et al. [60] reviewed some of the deep learning techniques used in machine health monitoring. Nowadays it is possible to generate enormous amount of data representing the operations of process machines due to availability of low-cost sensors for on-line measurements. Currently 'shallow' machine learning methods are applied in monitoring machine health related problems. But their application is limited to small data sets. Moreover, feature extraction in case of shallow machine learning algorithms is done manually, a procedure which is tedious and often ineffective in highly dimensional data. Deep learning algorithms are able to extract high level features thereby leading to further understanding of the current state of the process equipment. After training, deep learning models are able to distinguish between health and faulty equipment based on real-time on-line input big data. Deep learning algorithms which can be used in machine heath monitoring include DBNs, DBMs, CNNs, variants of RNNs and deep autoencoders. Figure 12 illustrates the use of deep learning techniques in machine health monitoring in relation to other alternatives.

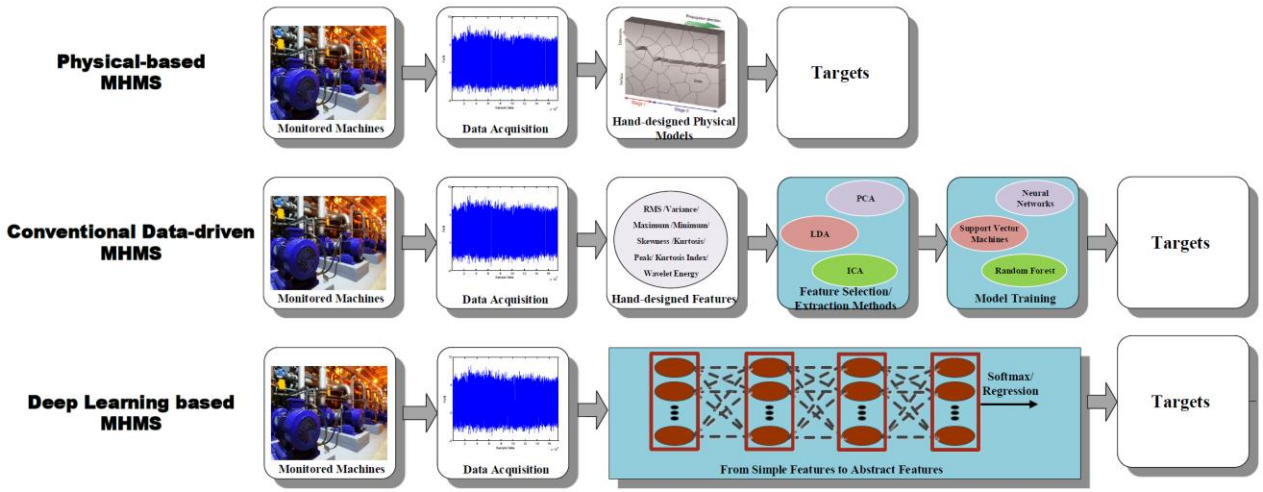

Figure 12. Three different models for machine health monitoring. [60]

In other studies, Deng et al. [61] proposed a deep learning method based on non-linear principal component analysis (kernel PCA) for process monitoring and fault detection. In their method, first a statistical model having a number of feature extracting layers is created. The statistical model includes both linear and non-linear PCA multiple layers with the aim of extracting both linear and non-linear intricate representations respectively and hierarchically. On top, Bayesian inference layers are added to transform the statistical data from the previous layers into fault probabilities. The probabilities are then weighted to obtain a probabilistic fault detection methodology. A schematic representation of their [61] proposed deep learning method is shown in Figure 13.

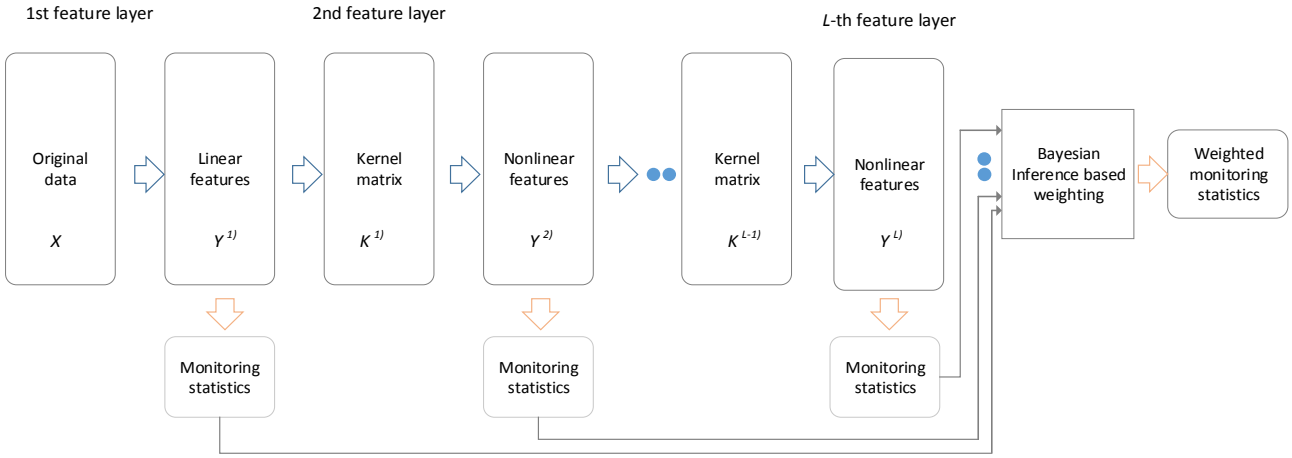

Figure 13. Deep learning with non-linear PCA. [61]

Deep learning is also increasingly applied in soft sensing technologies for industrial processes. For instance Yao and Ge [62] reported use of a semi-supervised deep learning technique coupled with hierarchical extreme learning for prediction of carbonmonoxide (CO) content from a process unit. The authors argued that currently soft sensing is limited by low-sampling rate of process output variables such as concentration of gaseous products. This can be partly attributed to complexity of processes like harsh process conditions and expensive component analyses. Therefore this leads to fewer labelled data available for development of soft sensors using supervised algorithms whereas the much larger unlabeled data is discarded. To address this pitfall, they applied unsupervised deep auto-encoder algorithms for feature extraction using all process data available. On top of that they employed a semi-supervised extreme learning machine (ELM) method as the last layer for regression. According to their findings, the proposed method performed far better than individual machine learning algorithms based on only ELM and hierarchical ELM. In a related work, Wang et al. [63] also successfully applied deep learning for a soft sensing method in a coke dry quenching process. Their ultimate task was to optimize the waste heat recovery in that particular process.

2.6.3 Deep learning software and tools

Implementation of deep learning models is computationally intensive. In other words, it would require high computation power to train complex deep neural network models in order to realize real-time process monitoring. However, due to the importance of new artificial intelligence applications such as image processing and machine translation, several commercial vendors offer platforms

where deep learning models can be developed, trained and executed in a considerably short periods of time. Many of these are accessible as platform as a service (PaaS) in cloud environment. Examples include IBM PowerAI platform, NVIDIA DGX-1 Software Stack, Intel Nervana platform, Skymind intelligence layer and so forth. There also many available open source deep learning libraries, which offer different deep learning models. Notable examples are H2O.ai, Tensorflow, Chainer, MXnet, Keras and many others.

2.6.4 Proprietary deep learning software and tools

Neural Designer

Neural Designer is a deep neural network commercial software developed by a start-up company called Artelnics from Spain. It is programmed in C++ language and applicable in tasks involving pattern recognition and predictive analytics. It was built using OpenNN, a deep learning framework that is currently available as an open source software library. Neural designer provides a user interface which enables the user to design his or her deep neural network without coding. It supports applications in science or engineering, business and health care. It is considered a desktop application software and can operate on computing platforms including windows, Linux and Mac OSX. Neural designer requires a data set as the input and its output is a neural model in form of a mathematical expression. The main activities performed during implementation of Neural Designer software are summarized in Figure 14. The software consists of three major components, Neural Editor, Neural Engine and a Neural Viewer. The editor allows the user to design the neural network as desired and permitted within the software. Performing from the background the Neural Engine allows the user to run tasks. In the Neural Viewer, the user can observe the tasks which have been run in forms such as texts, tables and graphical representations. [64]

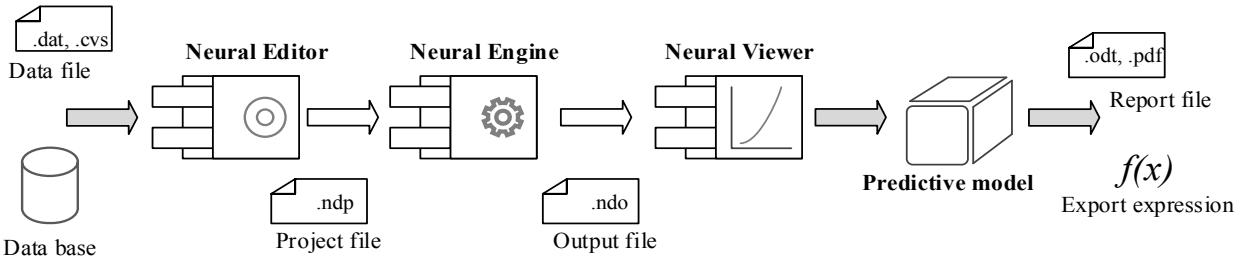

Figure 14. Schematic representation of main components of Neural Designer. [64]

Matlab Deep learning tools (from The MathWorks Inc)

There several deep learning tools available in latest versions of MATLAB® software like in MATLAB R2017a. These can be found under the Neural Network Toolbox™ in MATLAB®. It is also important to mention that others are externally available either as open source or premium and can be imported into Matlab and utilized as necessary. Some of these toolboxes include DeepLearn Toolbox, Deep Belief Networks (DBNs), Deepmat, Restricted Boltzmann Machines (RBMs), Deep Boltzmann Machines (DBMs) and ConvNet toolbox.

In Matlab the user can design, train and simulate different types of deep learning neural networks such as convolutional deep neural networks (CNNs) for image classification and autoencoders applied in feature learning. Generally, Matlab deep learning tools can be applied to problems related to classification, clustering, modelling and control of dynamic systems, dimensionality reduction and time-series prediction and so forth.

Training of deep neural networks (DNNs) with large datasets can be speeded up by leveraging Matlab' parallel computing toolbox which allows distribution of computations and data across multicore CPUs and GPUs on the desktop. In addition, computing capacity can be scaled up to clusters and clouds with the use of the Distributed Computing Server™. With small datasets, deep learning can be done by use of 'transfer learning' with pre-trained deep neural networks. [65]

Skymind intelligence layer (SKIL)

Skymind is an AI start-up company, which is behind the open-source deep learning library called Deeplearning4 (DL4J). The same company offers SKIL deep learning software as an open-source software for an enterprise. However, it can be integrated with proprietary vendor components necessary to run deep learning algorithms for different enterprise applications. SKIL combines capabilities of individual open source deep learning libraries such as DL4J, ND4J, DataVec, JavaCPP and LibND4J. SKIL software is applicable on data tools such as Spark, Kafka and Hadoop as shown in Figure 15. [66]

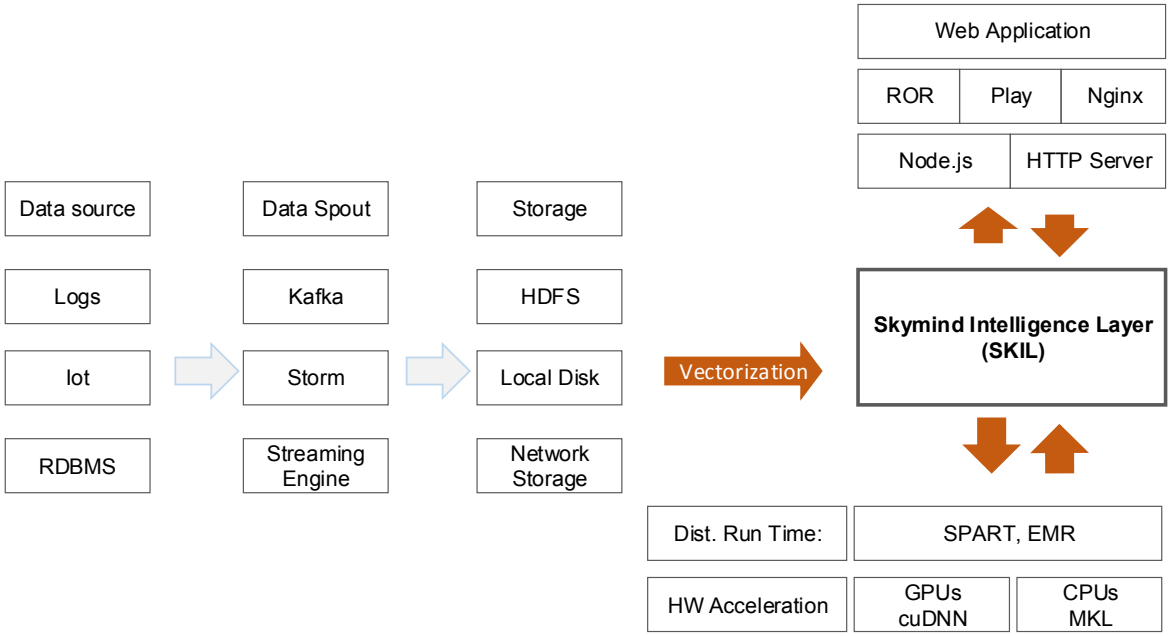

Figure 15. Skymind layout. [66]

IBM PowerAI Platform

IBM combined different deep learning frameworks and libraries which otherwise can be accessed individually as open source software to generate what it is called IBM PowerAI Platform. This plat-

form represented in Figure 16. It is argued that the platform makes machine learning and deep learning more accessible and offers better performance especially when used on IBM's Power Systems (e.g. IBM Power Systems S822LC for HPC server). It is designed with easy and rapid deployment in mind and suitable for enterprise operations. The software is applicable on Linux operating system (Ubuntu 16.04 LTS). [67]

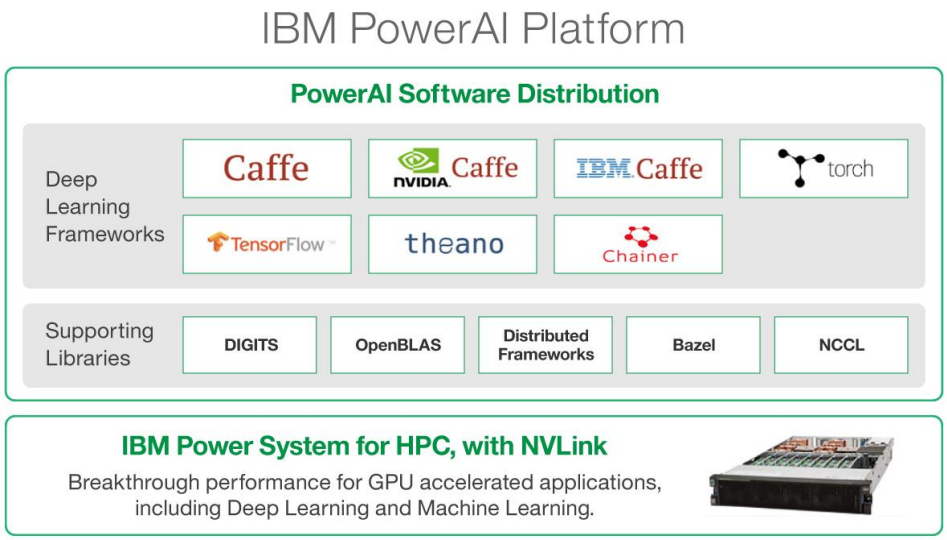

Figure 16. PowerAI Platform Stack. [67]

In early August 2017, IBM announced a new deep learning software called PowerAI DDL [68] aimed at reducing the time it takes to train models with huge data sets. It is noted that currently training complex deep neural networks can take several days or weeks in order to achieve desirable accuracy. With PowerAI DDL training of complex deep learning models can be achieved in few hours. This is achievable due to the capability of distributing individual tasks onto multiple computers. Hence, the name PowerAI distributed deep learning (PowerAI DDL). Before, deep learning mainly is executed on a single server due to the complexity and problems involved in transferring large data sets across multiple computers. A notable problem is the difficulty in data synchronization across different servers and processors. But PowerAI DDL offers a unified infrastructure for running deep learning algorithms across several GPUs for single nodes or multiple nodes and even in a cloud. PowerAI DDL is based on the integration of TensorFlow, Caffe and Torch deep learning libraries.

ViDi Suite 2.0

It is a deep learning based industrial image analysis commercial software offered by Cognex ViDi [69]. It is considered to be one of the first deep learning commercial software for machine vision. The software comprises of three individual image analysis tools, ViDi Blue, ViDi Red and ViDi Green and each of these perform specific tasks. For instance ViDi Blue searches and detects unique or complex local features within the image by learning. The Red tool is used for identifying anomalies after learning different variations of what is considered to be an acceptable appearance of the image. The Green tool learns the classification of the images based on the image labels. ViDi suite software is said to be easier to implement compared to conventional machine vision solutions. Training of the deep learning algorithms can be done with a single high-end NVIDIA GPU and this process is said to be fast. This software can be applied in a wide range of industrial operations such

as textile, pharmaceutical, automotive and printing. For example, in textile manufacturing through self-learning, defects in weaving patterns, dyeing and cutting can be identified by image analysis inspection. [69]

IBM Watson

IBM Watson is cognitive computing platform which combines artificial intelligence and advanced analytics. Mainly applied in cognitive business by offering business insights and also in healthcare in decision-making. In process industry, IBM's Watson IoT platform is used in visual inspection of product and equipment leading to improvement in product quality, availability, safety and maintenance. Like in other business areas, IBM Watson cognitive abilities can be used to offer guidance in decision making in process industry. A notable case is the use of IBM's cognitive solution software at Woodside Energy, an Australian oil and gas company. [70] Furthermore, recently IBM and ABB [15] announced a partnership aimed at realizing artificial intelligence in industrial processes, transportation, utilities and infrastructure. This is expected lead to fusion of IBM's expertize in AI technologies (such as cognitive computing and machine learning) and ABB's concrete knowledge in industrial automation solutions. Watson would offer real-time cognitive capabilities to process control and automation.

NVIDIA DGX-1 Software Stack

Nvidia provides NVIDIA DGX Software Stack, a hardware-software system for executing deep learning tasks and 'AI-accelerated analytics'. The system is illustrated in Figure 17. In this hardware-software package, Nvidia Digits, a deep learning software is integrated with other popular frameworks including Caffe, TensorFlow and Theano among others. According to Nvidia, the system offers fast training of deep neural networks using Nvidia Digits GPU. Complex deep neural networks can be created easily with the help of NVIDIA CUDA® API. The system also supports integration of cloud management services. [71]

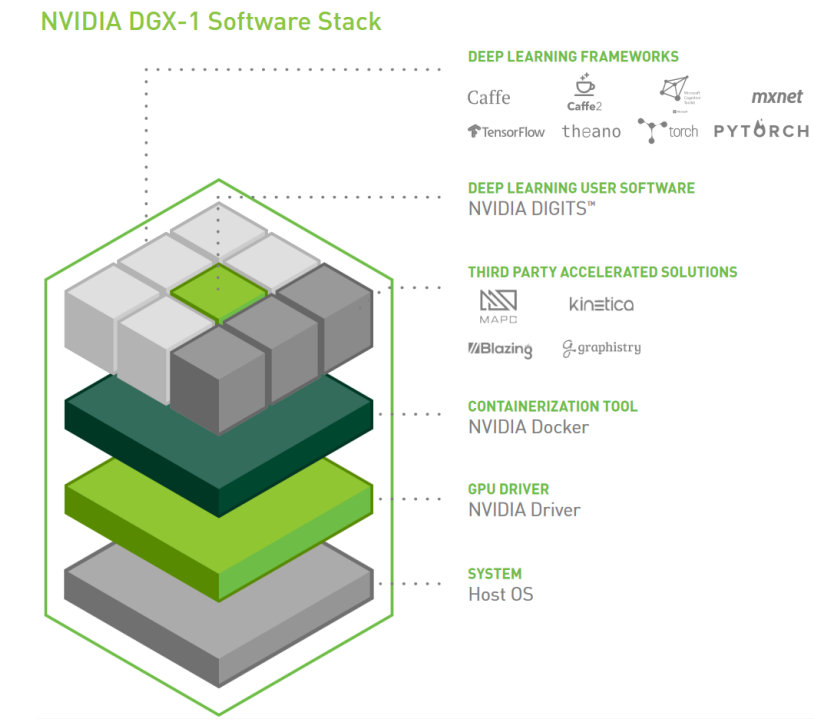

Figure 17. Nvidia DGX-1 Software Stack. [71]

Intel® Nervana™ Platform

The Nervana platform is also a commercial software and hardware platform used to create, design and implement deep learning algorithms. Its supports Windows, Linux and Mac OS operating systems. Although, Neon is the main framework, the platform also offers multiple deep learning frameworks and related libraries including TensorFlow, Caffe, BigDL, Theano and MXNet as illustrated in Figure 18. With the Neon framework, training deep learning models on large data sets can be realized in a short period of time, often in hours rather than days. The system is also scalable and flexible with several data types such as text, images, videos and time series being supported. The platform can be accessed either through the Nervana cloud or installed at the premises of the enterprise. Applications of this platform include predictive analytics, knowledge management, fraud detection, improvement of customer services in financial sectors and many others. [72]

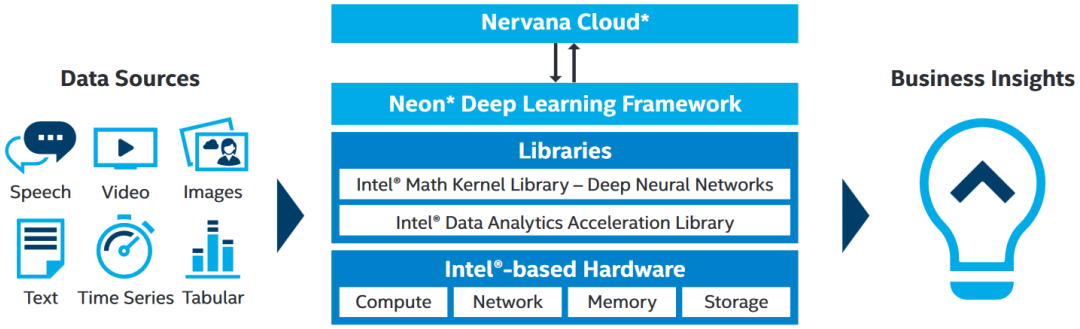

Figure 18. Intel Nervana platform stack. [72]

Wolfram Mathematica

Wolfram also provides the possibility to create, train and deploy deep neural networks especially in the field of computer vision, image processing and speech recognition. The deep learning framework is integrated within the wolfram language. Similarly to Matlab, deep learning tools can be accessible through the neural network framework. The framework supports both CPU and GPU training of deep learning networks. The user can leverage the 'NVIDIA CUDA Deep Neural Network library (cuDNN)' to enhance GPU performance. The commonly applied networks are convolutional neural networks and autoencoders. It is user friendly in the sense that coding is minimal with few lines of code often sufficient. [73]

2.6.5 Open source frameworks, libraries and tools

There are a number of deep learning software libraries, frameworks and other related software tools. Examples of these are briefly discussed below.

TensorFlow™

TensorFlow is one of the deep learning software libraries developed by Google Brain team. It is open sourced under the Apache 2.0 license. TensorFlow is useful in performing numerical computation with the help of data flow graphs. It is said to have a flexible architecture that allows the user to execute computational task on one or multiple CPUs or GPUs on desktop, server or mobile platforms with a single application programming interface (API). It can be applied on different programming interfaces which include Python, C/C++, Java, Go and R. Among other features, it offers automatic differentiation and supports CUBA, a parallel computing platform from Nvidia. However, it does not support open multi-processing (OpenMP). It is available for Linux, Mac OSX and Windows operating systems. Some community users have found it to be a too 'low level' thereby requiring more user input such as coding. [74-75]

Theano

Theano is a Python based open source software library developed by MILA research group at the University of Montreal. It is a compiler for mathematical expressions in Python. With Theano the user can specify, optimize, and solve mathematical expressions concerning multi-dimensional arrays on CPUs and GPUs. It is among the first libraries originally developed for handling extensive neural network algorithms associated with deep learning. Theano is applicable on different platforms or operating systems. It is a low-level library similarly to TensorFlow. Although it can be applied directly, often 'wrapper' libraries are added on top. Some light-weight libraries built on top of Theano, include Lasagne and Blocks. [76]

Keras

Keras is considered a high level library of neural networks API and supports convolutional networks and recurrent networks. It is written in Python and runs on top of other libraries, Theano, Tensorflow or CNTK. It is open source under the MIT license and operates on the Python or R interfaces. Keras is available for Linux, Mac OS X and Windows platforms. It is user friendly in the sense that minimal coding or user input is required. Keras developed on the guiding principles of user friendliness,

modularity, extensibility and its use of Python programming language. In Keras a model is described as a sequence or a graph of standalone, fully-configurable modules that can be combined together with less effort. Examples of standalone modules in Keras include neural layers, optimizers, cost functions, activation functions and initialization schemes, among others. Each module requires minimal coding. It is possible to add new modules, a feature which allows users to search for new ideas. [77]

Caffe

Caffe is a deep learning framework developed by Berkeley Vision and learning Centre. Like many others, it is open source under BSD license. It is a machine vision library written in C++ and can be implemented on Python and Matlab interfaces. Originally intended for image classification with the use of convolutional networks. But could be applicable in speech and multimedia processing. [78]

Microsoft Cognitive toolkit (CNTK)

This is similar to previous libraries, TensorFlow and Caffe and others like Torch. It can be added as a library in Python or C++ programs or applied as a standalone machine learning tool in BrainScript and is available for Linux and Windows operating systems. It is an open source software under the MIT license. The toolkit can be used to create, train, and evaluate user defined neural networks such as recurrent nets and convolutional nets either for research or industrial applications. The CNTK (computational Network Toolkit) was originally developed for speech recognition systems, this makes it suitable for handling time series data with use of recurrent neural nets. [79]

The NVIDIA Deep learning SDK

This a deep learning tool developed by NVIDIA and it is referred to as DIGITS. Only available freely for members of the NVIDIA Developer Program. DIGITS is employed in designing, training and visualization of deep neural network (DNNs) for image classification, segmentation and object detection. In the DIGITS Model Store, the user can download pre-trained models like AlexNet, GoogLeNet and LeNet. Model accuracy can be improved through hyperparameter sweep of learning rate and batch size configurations. It also allows the user to follow up with neural network training tasks, analyse accuracy and loss in real time. DIGITS provides an interactive environment and requires minimum coding thereby enabling the user to concentrate more on designing and training networks. [80]

Deeplearning4j

Deeplearning4j is an open source deep learning software library maintained by Skymind and the Deeplearning4j (DL4j) community, available under Apache 2.0 license. It is written for Java and Java virtual machine (JVM) but applicable on other programming interfaces such as Scala, Python and Clojure. It is a distributed deep learning framework, focuses more on industrial projects rather than academic research. Normally, it is combined with Hadoop and Spark and designed to operate on distributed CPUs and GPUs. Moreover, DL4j is able to process problems involving considerably large amounts of data within reasonably short time scales. For instance, its processing speed is comparable to that of Caffe when dealing with difficult image processing tasks but betters TensorFlow and Torch. [81]

Torch

Torch is a known scientific computing framework written in Lua programming language. It offers a variety of machine learning algorithms applicable in deep learning. It was developed by a group of researchers from New York University. It is available as open source software under the BSD license. Torch is said to be easy to implement through LuaJIT script language and the supporting CUDA platform. Like other frameworks mentioned earlier, Torch has neural network and optimization libraries which can be implemented efficiently over CPUs and GPUs. Companies such as Facebook, Google and Twitter are said to be using Torch in their activities. It is also widely used in research laboratories focusing on deep learning algorithms such as convolutional neural networks and deep reinforcement learning. [82-84]

OpenNN

OpenNN is a neural network software library maintained by Artelnics. It is written in the C++ programming language and is open source under the GNU Lesser General Public License (LGPL). The library provides a deep learning environment where the user is able to create neural networks with general approximation properties. OpenNN is a cross-platform software, it can be implemented on several available computing platforms. It supports parallel computing through open multi-processing (OpenMP) and CUDA application programming interfaces.

MXNet

MXNet is among the most popular current open-source deep learning frameworks that can be used to develop and train deep neural networks models. The Distributed Machine Learning Community developed MXNet. Several companies including Intel, Baidu, Microsoft, Amazon and Universities like Carnegie Mellon and MIT support the MXNet library. It is noted to be scalable and flexible, allowing the use of multiple programming languages such as C++, Python, R, JavaScript, Matlab, Go, Scala and Julia. Due to its 'mixed nature' or flexibility that allows use of both symbolic and imperative programming languages, operations such as tracking, debugging and modifying hyperparameters while working with deep neural networks become much easier. Moreover, it is supported by large cloud providers like Microsoft Azure and Amazon Web Services (AWS). Primarily developed to solve problems related to computer vision, speech recognition, natural language processing and computation of generative models, concurrent and recurrent neural networks. [85-86]

Chainer

Chainer is a standalone deep learning framework based on Python which is applicable in the implementation of complex neural networks. It was created by Preferred Networks a start-up located in Japan. Among the key features supported by Chainer include CUDA computation and multi-GPU functionality. Most deep learning algorithms are developed based on 'Define and Run' implementation criteria in which a computational graph is generated during the 'Define' phase and trained in the 'Run' phase. However, in Chainer these two stages are 'closely related'. This means that, the model's computational graphs are not constructed prior to its training but instead are 'implicitly memorized' as training takes place. In other words, with Chainer the history of computation is saved instead of programming logic. This kind of approach taken by Chainer is known as 'Define-by-Run'. This concept has been observed to increase memory usage efficiency. Chainer can be used to build

different types of network architectures, for example deep reinforcement learning, recurrent neural network models and variational auto-encoders. Toyota and Panasonic are among the companies employing Chainer. [87-88]

BigDL

BigDL is a distributed deep learning library applicable on Apache Spark (a distributed computing framework). Figure 19, shows illustrates the inclusion of BigDL in Apache Spark. However, it can also run on Apache Hadoop clusters. Mainly it supports using Scala or Python programming languages. It can be used for deep learning tasks for example those which require numeric computation. Also supports working with models from other deep learning frameworks including Torch and Caffe. It is scalable on Spark and thus applicable in big data analytics. BigDL also is considered to offer 'extremely high performance' with the help of Intel® MKL (Math Kernel Library) and the use of Spark. BigDL is suitable for writing deep learning algorithms for analyzing large data sets on the same Spark cluster. Another scenario is when the user would like to simply utilize the existing Spark or Hadoop cluster to run deep learning algorithms. [89]

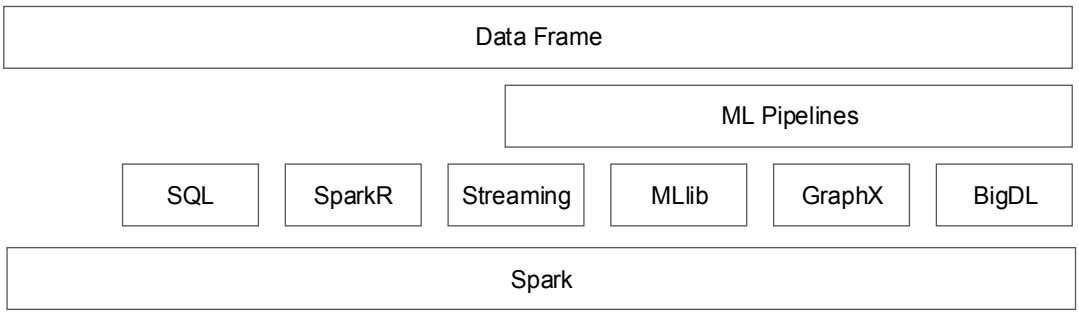

Figure 19. BigDL on Spark. [89]

Minerva

Minerva is another deep learning open source framework through the 'Apache V2' license. It employs a matrix-based application programming interface. According to the developers [90], it narrows the gap between 'productivity-oriented' and 'task-specific' software tools for instance Matlab and Caffe respectively. It can be implemented on both CPUs and GPUs. It utilizes model and data parallelism features enabling fast computation times and systems' scalability. Programming can be done in C++ or python languages and also has coding style similar to Matlab. Deep learning algorithms which can be executed in Minerva include CNNs and LSTMs. [90-91]

H2O.ai

H2O is an open source machine learning and deep learning library. In addition to deep learning, there are other in-built algorithms like Boosting and Bagging which the user can apply to design 'smarter applications'. Notable other machine learning algorithms present in H2O include regression tools, time series, k-means, random forest and so forth. Thus, it can be used for supervised and unsupervised data training tasks. It is fast, scalable in handling large data sets due to in-built memory

compression and distributed processing of data. Moreover, it is built on top of Hadoop and Spark clusters as shown in Figure 20. The H2O platform is written in Java interface but also supports use of programming languages such as R, Scala and Python. [92-93]

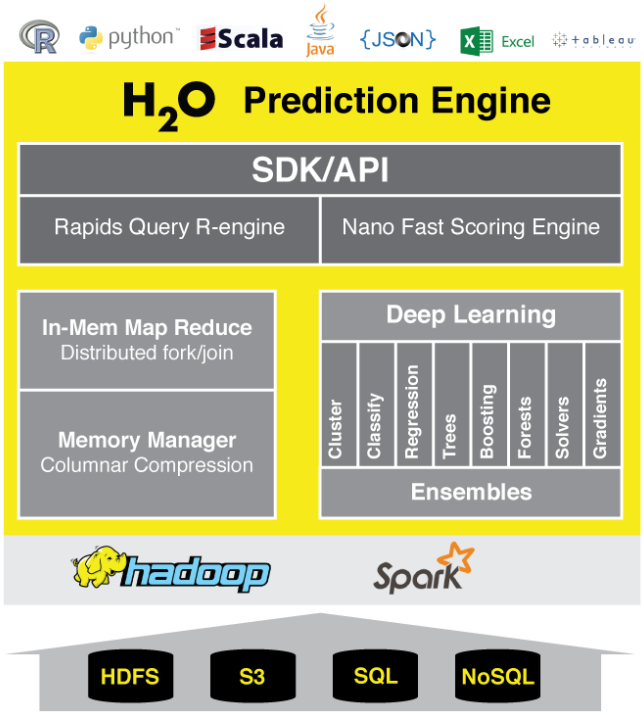

Figure 20. H2O Software stack. [94]

Neon

Neon is an open source deep learning framework developed by Intel Nervana. Like many others above, latest versions are several times a year and offer varying features. It is written based on Python and can be implemented on CPUs and GPUs. Deep learning models which can be developed in Neon include CNNs, RNNs, auto-enconders, LSTMS and reinforcement learning models. The software package also includes Intel Math Kernel Library (MKL) support for CPUs, which is useful in 'multi-threading' tasks. The use of neon on GPUs would require installation of CUDA API. Nevertheless, Neon also supports GPU architectures such as Pascal, Maxwell and Kepler. [95]

MatConvNet

MatConvNet is an open source library for convolutional neural network under the 'BSD-like' license. [96] It is applicable in the MATLAB interface. The toolbox is used for creating CNNs using MATLAB functions, offers routines for executions of convolutions, pooling, normalizations and so forth. It supports computations on both CPUs and GPUs. Like many other software tools above, it is applicable in training complex models on large data sets. However, MatConvNet is limited to implementing CNNs and thus it is more useful in cases where CNNs are effective, for instance in image classification, video recognition and natural language processing. [96]

Purine

Purine is also an open source deep learning framework. It is built based on bi-graph networks with tensors and operators, similarly to Caffe. The bi-graph is computed from sources to sinks with the help of so-called task dispatcher. The bi-graph can be computed in parallel over a cluster CPUs or GPUs. [97]

2.7 Applications of state-of-the-art AI methods: process industry cases

Artificial intelligence in process industry can be found in process control, monitoring, prediction analytics and so forth. However, AI use in process industry is currently dominated by traditional AI methods such as expert systems and various machine learning methods. Therefore, it is fair to say that there not many process industrial cases where big data and deep learning algorithms are combined and utilized in practice. Most of the information available rather point to the future of different process industries in this new era of AI and big data. Research concerning AI application in process industry has been done and much more is going on within various industrial process fields such as fine chemicals, metals and mineral processing, oil and gas, food processing, among others. A few industrial cases where new AI methods have been reportedly practiced are discussed in this section.

2.7.1 Chemical industry

In 2015, Mitsui Chemicals and NTT Communications successfully predicted the composition of gas products at one Mitsui's Chemical plants using deep learning algorithm. The pilot testing work is demonstrated in Figure 21. The method was implemented by modeling the relationships between various data sets. Among these included composition of feed materials to the reactor, operational conditions of the reactor (like temperature, pressure and flowrates) and the residual impurities in the gas product. The model reportedly showed quite good prediction accuracy with discrepancies only in the limits of ±3%. The deep learning algorithm was able to predict the chemical quality of the gas products within 20 min prior to generation of the final product. This method can be applied in detection of faulty sensors, monitoring product quality as well as prediction of future process conditions. [98-99]

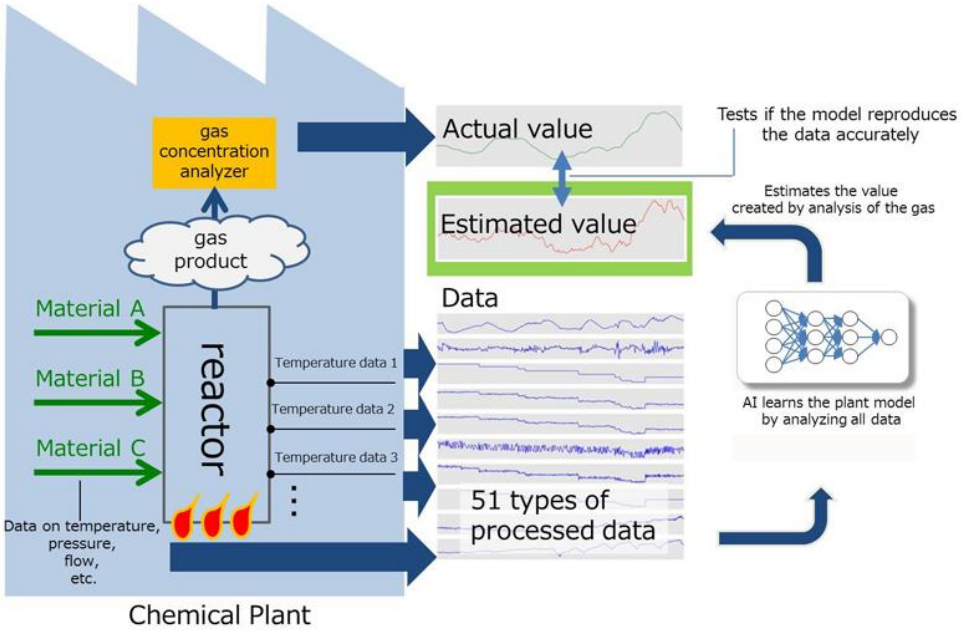

Figure 21. NTT Chemical plant scheme. [99]

2.7.2 Metals processing industry

Big River steel mill in the U.S., Arkansas is thought to be the 'world's first smart steel production facility'. It was commissioned in late 2016. The mill applies AI platform known as BEAST (Best Enterprise AI Supercomputing Technology) developed by Noodle.ai, a start-up founded in 2016 and located in the Silicon Valley. The mill was built with a significant number of sensors thus able to generate huge amounts of data, which is then processed by a supercomputer for data analysis. The AI platform at Big River comprises of predictive AI and machine learning algorithms. It is said to have been developed by experts in the AI field and some of whom were also involved in the development of IBM's Watson supercomputer. The actual architecture of this AI platform could not be found in literature. But it has been compared to self-driving cars, designed to learn and improve in performance with experience being on the road. In the same way, AI technology at Big River steel mill is based on learning from experience as the mill continuously melts and rolls tons and tons of steel over a longer period. In other words, the technology takes advantage of the larger sets of sensor data and monitor different variables for example heat, pressure, metal composition and equipment wearing, to further optimize production and maintenance with minimum manual input. Other areas of improvement include product quality, production planning, scheduling, logistics operations and environmental protection. [100-101]

Noodle.ai platform is available as a service referred to as 'AI-as-a-Service' [102]. The users need to subscribe monthly to particular predictive AI engines in order to utilize the services which are provided in the cloud environment. The subscription covers the costs associated with tasks such as customizing and deployment of the AI model for the customer, including model configuration, training and tuning plus other services related to new updates from Noodle.ai. The Noodle.ai platform is applicable in areas such as maintenance planning, production planning and logistics among others.

In another development, since early 2017 POSCO, a steel manufacturing company is currently using AI technology at Gwangyang Steel Works galvanizing plants aimed at controlling the amount of coating added onto steel. The artificial intelligence method applies big data and deep learning algorithms to control the zinc coating processes. The software was developed by POSCO Technical Research in collaboration with Sungkyunkwan University, Korea. Previously this process was manually controlled and thus large discrepancies in the amount of coating added was inevitable. Moreover, any changes in operating parameters also affected the coating weight control. But with the employment of the 'self-learning' control method, optimum amounts of coating were achieved even with upsets in the production process. [103]

On the engineering side, CeleraOne GmbH in Germany provides the steel industry with surface inspection, machine learning and predictive analytics technologies. Deep learning algorithms are utilized in the surface inspection tools. [104] Eigen Innovations in Canada, also reported a data analytic platform which applies deep learning tools for monitoring blast furnace operations in real-time. The platform has been reportedly tested at Glencore Xstrata. [105]

3 Cloud computing

According to Amazon [106], cloud computing is the 'on-demand delivery of computer power, database storage, applications, and other IT resources through a cloud services platform via the internet with pay-as-you-go pricing'. In other words, cloud computing is outsourcing computer services from software programs, applications to data centres via the internet [107]. Among the benefits of cloud computing, the computing resources are easily scalable, users only pay for what they use and with a self-service access to all offered IT resources. Furthermore, the user doesn't need to worry about important things like data storage and computing power since they are provided as services. Clouding computing is categorised into three main models which include infrastructure as a service (IaaS), platform as a service (PaaS) and software as a service (SaaS) [106-108]. The services provided by each model are illustrated in Figure 22 in comparison with the traditional computing platform.

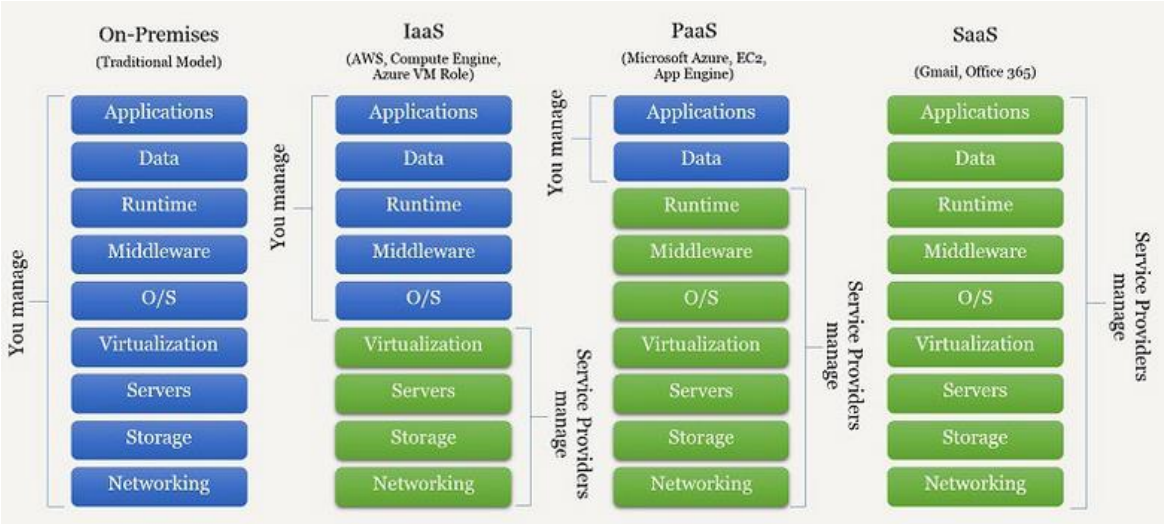

Figure 22. Comparison of traditional and cloud computing models. [109]

3.1 Infrastructure as a Service (IaaS)

Infrastructure as a service (IaaS) offers computing, data storage and networking infrastructure resources to clients. For instance, computing resources include virtual machines (VMs), network connections, operating systems, access to the network, load balancing and so forth. With the IaaS, the clients can utilize the computing resources to install their own middleware and software applications. The users who are often system administrators can also install their preferred choice of operating system or apply what is available from the cloud service provider. In addition, the service provider manages the infrastructure. Examples of IaaS include Amazon Web Services (AWS), Microsoft Azure, OpenStack, Rackspace, CloudStack, Citrix and Google Compute Engine.

3.1.1 OpenStack

OpenStack is a popular open source infrastructure as a service cloud computing platform which is available freely under the apache 2.0 license. It provides virtual resources for computing, storage, identity, networking, image services, etc. Clients who are typically cloud architects or administrators leverage the large pool of virtual resources through a self-service access, to create and manage either private or public clouds. OpenStack works with Hadoop for big data needs, scales vertically and horizontally to meet diverse computing needs, and offers high-performance computing (HPC) for intensive workloads. Key features include VM image caching, role based access control, VM image management, LAN management, VNC proxy via web browser, floating IP addresses, and much more. OpenStack architecture comprises of different projects namely Nova, Neutron, Swift, Cinder, Glance, Keystone, Horizon, Ceilometer and Heat. Each of these projects contribute to one or more of the core services of OpenStack demonstrated in Figure 23. [110]

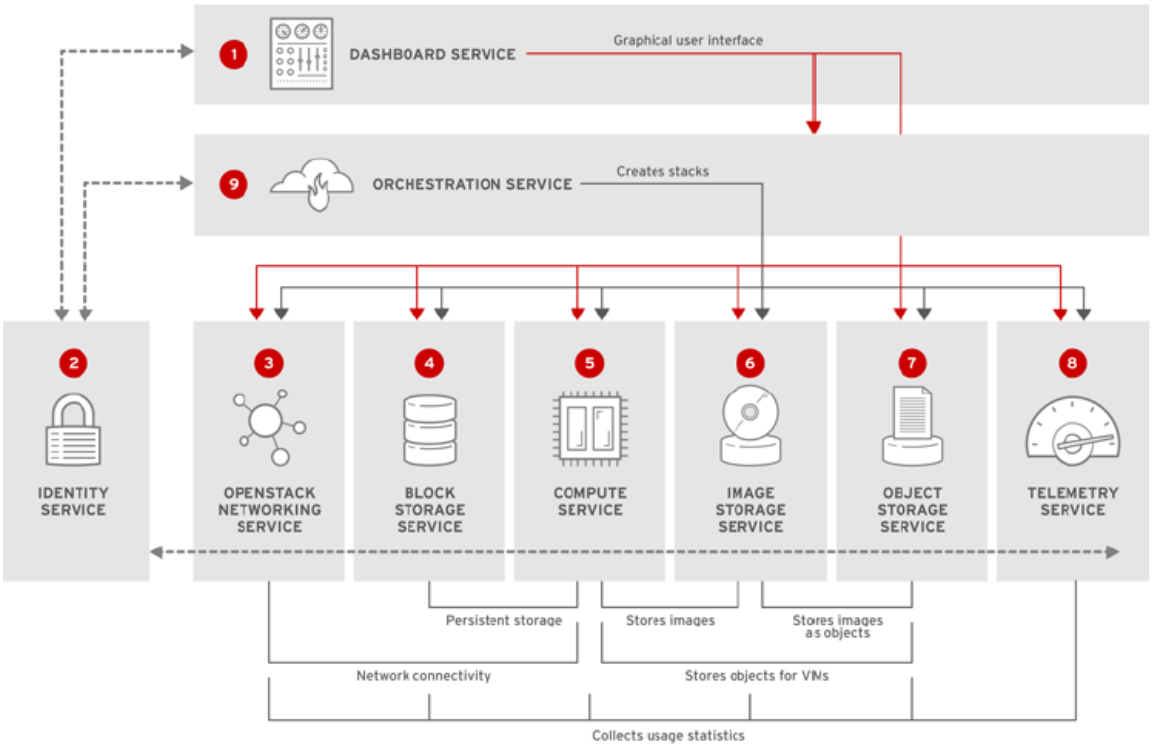

Figure 23. A simplified overview of OpenStack architecture. [110]

For instance, Nova handles the core of OpenStack computing services which involves managing and provisioning of virtual machines running on hypervisor nodes. Another notable component is Keystone which is responsible for identity or authentication services.

3.1.2 Apache CloudStack

CloudStack [111] is also a notable open source software for IaaS cloud computing which is managed by Apache Software Foundation (ASF). Similarly to OpenStack, it provides free software tools for deployment of either private or public clouds. It is said to offer an 'open and flexible cloud orchestration platform' for realizing reliable and scalable private, public or hybrid cloud environments. To run IaaS cloud, CloudStack employs a management server and supported hypervisor hosts (e.g. BareMetal, XenServer, Hyper-V, KVM, etc.) as illustrated in Figure 24. CloudStack provides a scalable infrastructure, for instance it can manage many physical servers located in widely remote distributed datacenters. Moreover, issues such as maintenance of the management server may take place without interrupting the performance of virtual machines. Users like administrators manage the cloud through a Web interface and REST-like API. Other notable features include compatibility with Amazon's EC2 and S3 cloud tools.

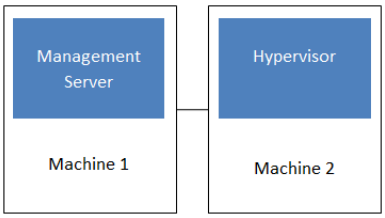

Figure 24. A simplified basic deployment on CloudStack. [111]

Figure 25 provides an overview on how resources in CloudStack are arranged. Highlighted are Regions, Zones, pods and clusters. A region comprises of one or more geographically nearby zones and managed with one or more management servers. On the other hand a zone represents a single data center and it may consist of one or multiple pods and a secondary storage. Typically a pod can be a single rack or row of racks with one or more clusters. As demonstrated in Figure 25, a cluster comprises of one or several homogenous hosts (or hypervisor/computer) in addition to a primary storage.

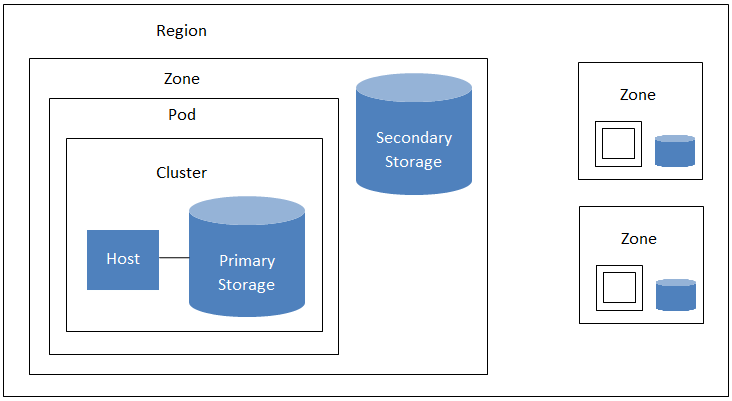

Figure 25. CloudStack infrastructure overview. [146]

3.2 Platform as a Service (PaaS)

Platform as a service (PaaS) is a model layer on top of IaaS. Components of the PaaS layer include application servers, enterprise service buses, database management systems, rule engines, business intelligence or analytics and so on. The PaaS offers the users with resources for developing, testing, deployment and execution of software applications onto the cloud infrastructure. PaaS supports collaborative work amongst different teams. Management of the underlying infrastructure is mainly the responsibility of the service provider. This enables clients to focus more on application development. Service providers also handle issues related to security, operating systems, server software and back-ups among others. The user can manage deployed applications such as making changes, configuration settings and managing updates. The users of PaaS are mainly application developers for instance enterprises with internal software development programs and private software developers or independent software vendors (ISVs) in SaaS business. Examples of PaaS include, Microsoft Azure, AWS Elastic Beanstalk, Cloud Foundry, OpenShift, CloudBees and Google App Engine.

3.2.1 Cloud Foundry

Cloud Foundry is an open source cloud platform as a service available under Apache 2.0 license. It was originally developed by VMware before being managed by Pivotal Software. With Cloud Foundry developers are able to create, deploy, execute and scale applications on public, private or hybrid cloud environments. Cloud Foundry offers developers multiple choices of programming languages (e.g. Java, PHP, Python, Ruby, Node.js, and Go) and framework support, features which limits the applicability of other cloud platforms. In other words, Cloud Foundry offers an extensible architecture which prevents undesirable circumstances such as vendor lock-in and also permits future modifications. It supports application on different cloud infrastructure as a service (IaaS) like VMware, AWS, OpenStack, vSphere, IBM, Azure and so forth via a BOSH service layer. At the moment, Cloud Foundry is considered as the industry standard for PaaS. Hence, there a number of industry grade cloud platforms (PaaS) based on Cloud Foundry architecture. For example, IBM Bluemix, Pivotal Cloud Foundry, NTT Enterprise Cloud, GE Predix, Baidu Cloud, SAP Cloud Platform and MindSphere among others. Figure 26 describes a simplified architecture of Cloud Foundry platform. [112]

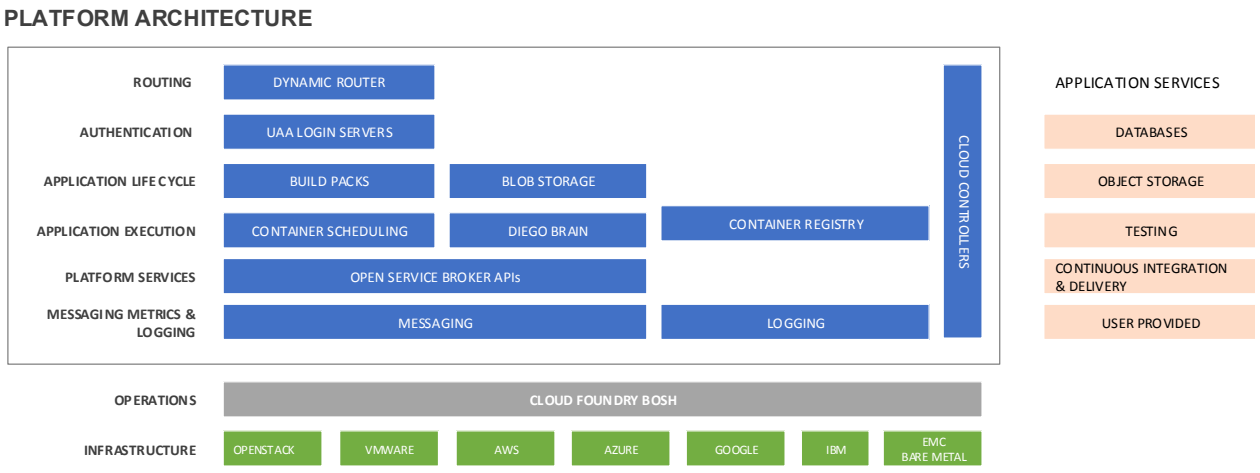

Figure 26. Cloud Foundry platform as a service architecture. [113]

3.2.2 OpenShift

OpenShift is a notable competitor of Cloud Foundry. It is an open source container application platform as a service from Red Hat where Kubernetes and Docker are utilized. It is also available under Apache 2.0 license. There variants of OpenShift cloud platforms but OpenShift Origin is the overall community project where developers are able to create, test, deploy and execute container-based applications. The other OpenShift platforms are built using OpenShift Origin as the upstream code and for that reason, here only OpenShift Origin is briefly discussed.

OpenShift Origin can be described as a 'distribution of Kubernetes optimized for continuous application development and multi-tenant deployment'. Within the OpenShift Origin architecture, developer and operations tools are added on top of Kubernetes to achieve faster development of applications, effortless deployment and scaling as well as long-term life cycle maintenance. Figure 27 shows the architectural overview of OpenShift Origin highlighting the key components such as the routing layer, master, the nodes (containers) and the infrastructure layer. Moreover, similarly to Cloud Foundry, several programming language interfaces are supported (e.g. Ruby, Python, Node.js, PHP, Perl and Java) and applicable on different cloud services. [114]

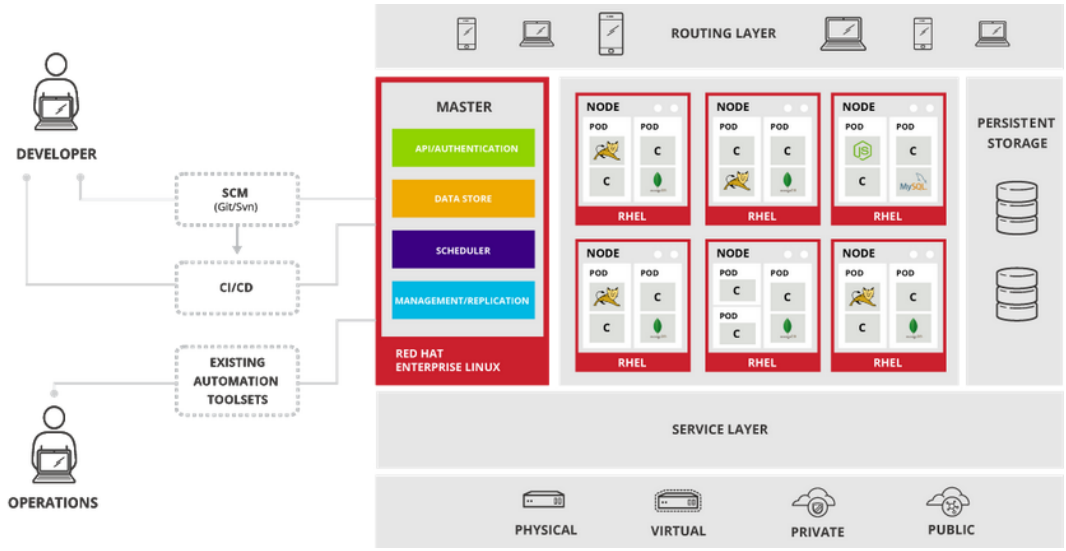

Figure 27. The overview of OpenShift Origin architecture. [114]

3.3 Software as a Service (SaaS)

Software as a Service provides a complete application package from the cloud service provider (Figure 22). It offers a wide range of software applications and tools for the user to leverage including services offered by IaaS and PaaS platforms. In SaaS model the cloud service provider manages the application domain as well as the rest of the software and hardware stack on which the applications are built upon. SaaS is mainly offered via the internet or web browser interface to the consumers but in some cases through employment of application programming interfaces (APIs) to the cloud service. Examples of SaaS are Salesforce, Google App, NetSuite and Concur.

3.4 Industrial cloud platforms

There are several cloud platforms available for use in industrial internet of things and big data analytics. Key players include cloud services providers (like Microsoft Azure, Amazon Web Services, IBM, Intel etc.), enterprise solution vendors (like PTC and Oracle), networking companies (such as Cisco and AT&T) and industrial engineering companies (e.g. GE, Siemens and Bosch) among others. Many of the platforms are available under proprietary licenses with a few others being accessible as open source projects. Some of these platforms are presented here, with much emphasis put on the best available open source cloud platforms that can be employed in connecting 'things' such as sensors, actuators among other devices to the cloud or internet. This kind of connection is demonstrated simply in Figure 28.

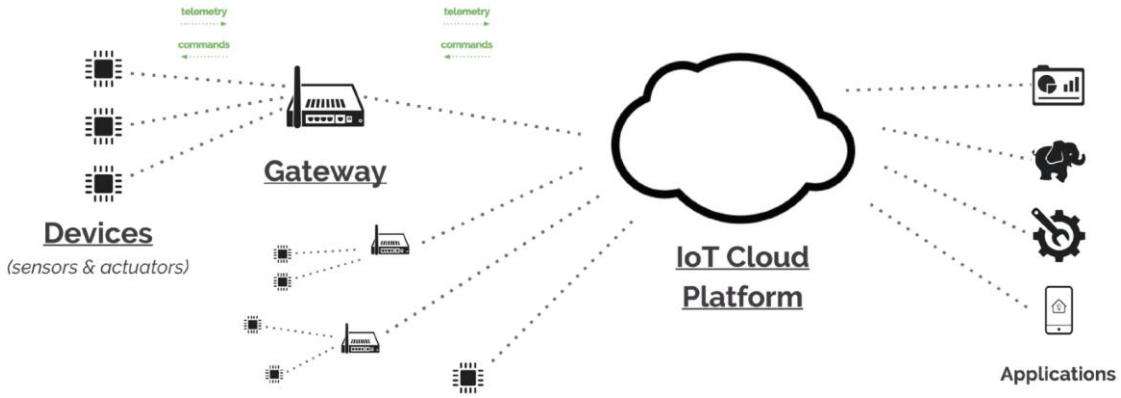

Figure 28. A simplified IoT architecture. [115]

3.4.1 Proprietary industrial cloud platforms

As previously noted, cloud platforms available from highly recognised cloud service providers, networking firms and industrial automation companies. Here only a selected few are briefly discussed.

Amazon Web Services (AWS) IoT cloud

The AWS IoT [116] is a well-known commercial IoT cloud platform supported by Amazon Web Services. It provides a secure two-way communication between connected things (assets) or devices and the AWS cloud. It offers collection, storage and analysis of data from multiple devices. Also, it allows interaction between applications and connected devices. It is possible to develop applications which allow users to control the devices by using smart phones or tablets. The architecture of the AWS IoT cloud comprises of the following components.

First, the AWS IoT Device SDK which provides the tools to easily and quickly connect hardware devices or mobile applications. It allows connection, authentication and exchange of messages with AWS IoT with the use of MQTT, HTTP, or WebSockets protocols. Here supported programming languages include C, JavaScript and Arduino. The use of this feature can be replaced with any other appropriate SDK (software development kit) alternative.

Another component is the Device Getaway, which enables a secure communication between the devices and the AWS IoT platform. Messages can be exchanged via the Device Gateway with use of the publication/subscription model and allows 'one-to-one or one-to-many' information sharing. The MQTT, WebSockets, and HTTP 1.1 protocols are supported.

The message broker, this ensures a secure exchange (publishing and receiving) of messages between devices and AWS IoT applications. Here MQTT protocol or MQTT over WebSocket can be used to subscribe and for publishing messages. The HTTP REST interface can be applied for publishing tasks.

The identity of devices is handled by the Registry which also allows storage of device's metadata like device attributes and capabilities. Each device is assigned a unique identifier. There is no expiry date for the metadata in the registry however if it has not been accessed or updated at least once in seven years, information will expire.

The AWS IoT offers a possibility to design a 'persistent', device shadow (virtual version) of individual devices such that applications and other devices can read messages and communicate with the device. The device shadows persist the last reported state and desired future state of each device even when the device is offline. Information can be retrieved for the previously recorded state of a device or its desired future state with use of the APIs or the rules engine. Developers or users can build applications capable of interacting with the devices using device shadows. This task requires giving the available REST APIs. Moreover, the applications can set a new desired future state of the device without basing on the latest state. In this case, the AWS IoT makes comparisons between the desired and the previous state and after commands the device to make necessary changes. With device shadows, the state of the device can be stored freely up to one year. Device shadows have no expiry date but need to be updated at least once a year, otherwise they cease to be valid.

The other component is the rules engine which allows creating IoT applications for instance for collecting and analysing data from connected devices. It is also applied in evaluation of incoming messages into AWS IoT and handles messages transformation and delivery to other devices or cloud services as the per user's rules. The rules engine is also used in routing messages across AWS endpoints such as AWS Lambda, Amazon Kinesis, Amazon S3, Amazon machine learning and so forth.

Also a notable component of the AWS IoT architecture is the 'Security and Identity service' which literally handles the security aspects in the AWS cloud. This includes ensuring that credentials of each device are secured while sending and receiving messages or data. The AWS security features are utilized by both the message broker and rules engine to securely transmit data in between devices and AWS services. Figure 29 illustrates the functions of the key components of the AWS IoT platform.

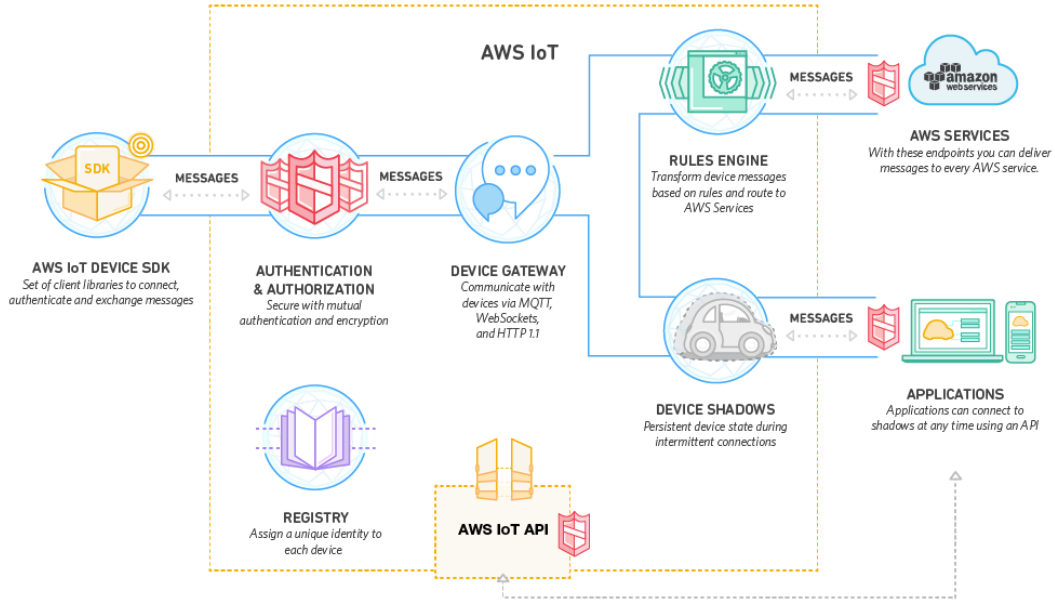

Figure 29. Architecture of the AWS IoT cloud platform. [116]

Microsoft Azure IoT Platform

Microsoft Azure IoT Suite also covers the core areas of internet of things cloud computing such as device connectivity, data processing, analytics and management as well as presentation and business connectivity. The connection of devices to the cloud is done either directly (via internet) or indirectly (for instance use of industry specific protocols like CoAP, OPC UA) through the IoT gateway and in both cases edge intelligence maybe employed alongside some processing capabilities. As noted in the previous Amazon platform, the getaway provides a two-way communication channel between the front and back ends. The back end features include device registry and discovery, data collection, transformation, data analytics plus business logic and visualizations. Microsoft Azure IoT Hub is implemented as the layer for Cloud Gateway, in other words connecting the devices to the cloud interface. Figure 30 illustrates how Microsoft Azure platform can be implemented in industrial automation. [117-118]

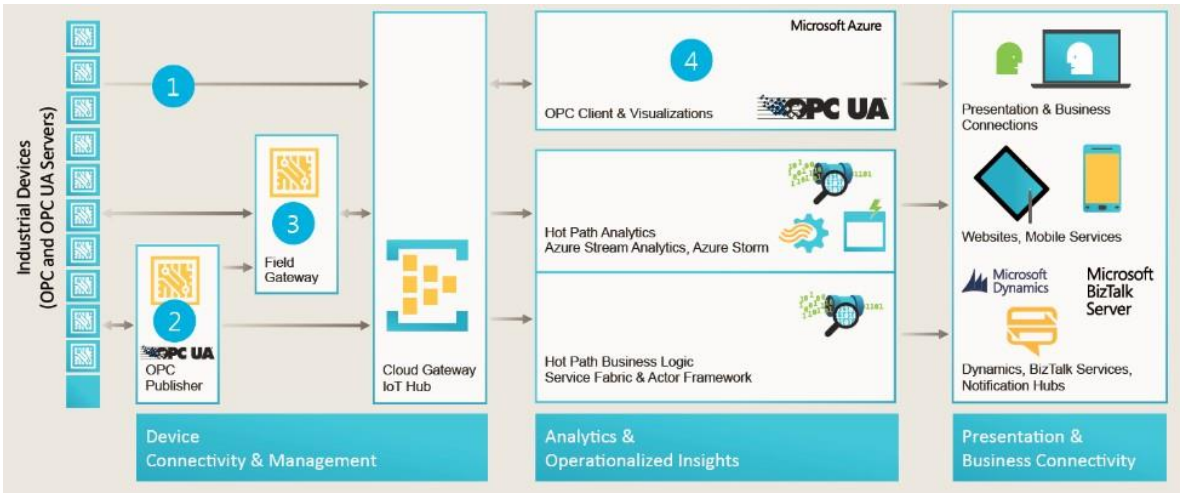

Figure 30. Illustration of Microsoft Azure Industrial IoT Platform. [118]

Predix platform

Predix platform [119] is an industrial cloud based platform available from General Electric (GE) firm as Platform as-a-Service. With the Predix platform, machines or devices, data as well as people can be connected using state of the art computing technologies. It offers distributed computing, big data analytics and device data management tools in addition to machine-to-machine interaction. The Predix platform is designed for industrial IoT applications and thus it can be rapidly applied in development of industrial applications. The main components of the Predix platform include Predix Machine, Predix Services and Predix Cloud. The Machine layer installed on the getaways, industrial controllers, sensor among other things and it is responsible for collection of data from devices and sending it to the Predix Cloud. Machine layer also performs other tasks for instance edge analytics. One of the purposes of the Predix service layer is to enable developers create, test and deploy industrial internet applications. The Predix Cloud is secured and is well optimized for industrial workloads. It is also important to note that, the Predix platform is built on top of Cloud Foundry platform, which is an open source Platform as-a-Service. The employment of Cloud Foundry in the Predix platform architecture is attributed to provision of benefits such as easy management of application lifecycle, centralized management system for applications, distributed environment and ease of maintenance. The Predix platform architecture is illustrated in Figure 31.

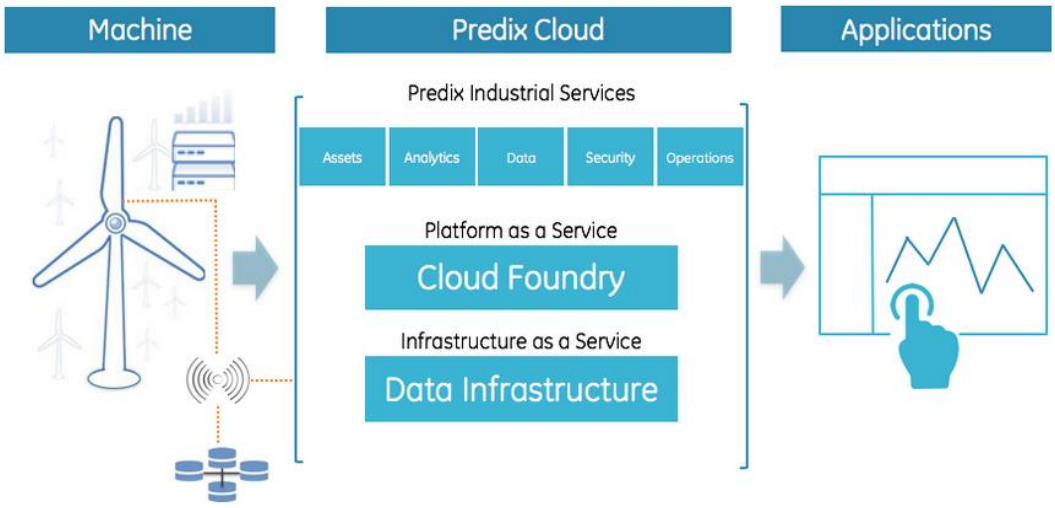

Figure 31. Predix platform architecture. [119]

MindSphere IoT Platform

MindSphere is a middleware cloud-based operating system platform for industrial IoT from Siemens. It is available as Platform as-a-Service (PaaS) for connecting devices, data analytics and also provides tools for developers to design their own applications. MindSphere is designed to assist in achieving better industrial plant efficiencies by leveraging information derived from large sets of data. The architecture of MindSphere IoT platform is composed of three key elements which include MindConnect, MindSphere and MindApps. Data from devices is collected by MindConnect layer which then securely sends it to the MindSphere middleware. In MindConnect, machines can be connected in several ways such as 'plug and play' or use of standard industrial automation architec-

tures such as OPC UA. MindConnect employs IoT gateways like MindConnect Nano and MindConnect IoT 2000 which offer a secure information sharing. The MindSphere layer is an open interface and allows users to develop applications specific to their demands. MindSphere layer is built on Cloud Foundry as observed with Predix. But other cloud platforms can be used to implement the MindSphere operating system. [120]

3.4.2 Open source industrial IoT platforms

Kaa IoT Platform

Kaa is one of the currently available multi-purpose open-source IoT platforms applicable in industrial settings. It provides the necessary IoT tools for connecting sensors, controllers, machines, device gateways among other things to the cloud environment. It supports interoperability and unified data sharing in between connected things. Furthermore, Kaa offers features which can be implemented in designing end-to-end applications for purposes of industrial automation, predictive maintenance and remote monitoring [121]. Some of the key features of Kaa IoT platform include device management, data collection, configuration management, messaging functionalities, edge computing among others. [122]

DeviceHive IoT platform

DeviceHive is an open source IoT platform which also can be used to collect data from devices. It is available for use either on public or private cloud services. It is scalable from single virtual machines to clusters. DeviceHive supports use of big data tools such as apache Spark, ElasticSearch, Cassandra and Kafka. It can be applied in areas of data storage and analytics, production control and industrial automation as well as predictive maintenance. [123]

Mainflux IoT platform

Mainflux is another open-source IoT cloud platform written in Go programming language and like many others, it's developed based on a set of microservices. It supports connections of devices and applications over a wide range of network protocols such as via HTTP, MQTT, WebSocket and CoAP. This means that, different network protocols can be implemented within the platform. Moreover, any device can be connected to Mainflux because it uses a generic device representation. It can be used to design complex industrial IoT solutions. It comprises of a device manager, application manager, messaging bridge and a distributed database. The device manager allows device connections on the 'southbound' interface whereas the application manager caters for application connections on the 'northbound' interface. The messaging bridge relays messages across devices and applications. Figure 32 demonstrates a simplified Mainflux IoT platform. It can be said that detailed features of Mainflux are similar to other IoT cloud platforms. [124-125]

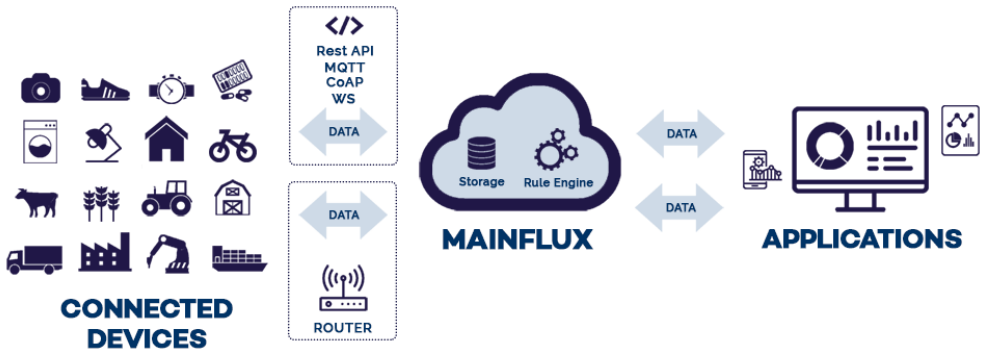

Figure 32. Mainflux IoT platform. [125]

Eclipse Kapua IoT platform

Eclipse Kapua [126] one of the many open source projects under Eclipse IoT group and supported by Eurotech as well as Red Hat. It is a modular, integrated and interoperable IoT cloud platform which can be implemented in collection and management of devices to other IoT cloud services. This means that, it provides the necessary tools to manage IoT gateways and smart edge devices within an integrated framework. Among the core IoT services available in Eclipse Kapua include, device registry, device management, messaging, data management and application enablement. The connection of devices to IoT gateways is supported by MQTT protocols. The connectivity layer also serves the purpose of device authentication and authorization. In general, other details of the platform are comparable to previously discussed platforms. The commercial version of Eurotech industrial IoT platform is known as Everyware Cloud. This cloud is developed based on Eclipse Kapua and Kura open source platforms [127].

Lelylan IoT platform

Lelylan is also an open source IoT platform available under Apache 2.0 license. It was developed based on independent microservices with high flexibility and scalability. This implies that developers are free to add or even replace some of the microservices without affecting the rest of the platform. Hence, the platform can be easily incorporated with other IoT functionalities such as big data software tools. The Lelylan platform can be implemented on other cloud services, either on public or private datacenters. Currently there are efforts to improve the deployment process with the help of tools like Docker, Mesos and Ansible [128]. The platform's architecture overview is demonstrated in Figure 33.

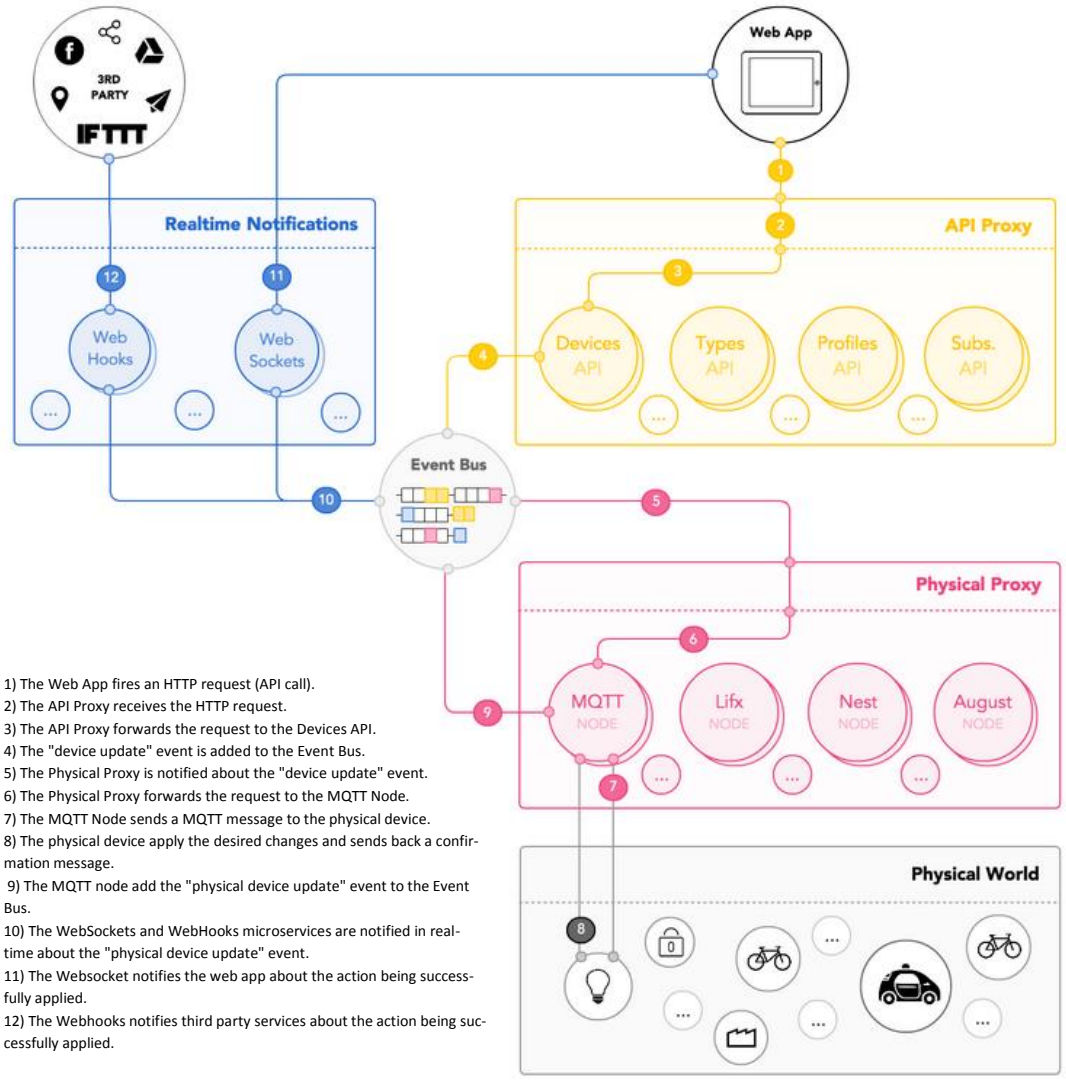

Figure 33. Architecture of Leylan IoT cloud platform. [128]

WSO2 IoT platform

The WSO2 IoT server is an open source platform under the Apache 2.0 license. It is a comprehensive and flexible platform applicable in industrial IoT middleware layer. The WSO2 server also allows connection of mobile devices to applications and managing them securely. In addition it offers real-time, batch and edge data analytics. [129]

SiteWhere IoT plaform

Sitewhere is an open source platform under the CPAL 1.0 license. It allows data collection from devices, offers data storage and processing capabilities as well as device management and device integration tools. The IoT sever can be either installed on premises or in the cloud. The core of the Sitewhere platform is built based on open source technologies such as Apache Tomcat 7, Spring Framework, Spring Security and Hazelcast. Also applies data processing tools like Apache Spark, InfluxDB and Apache HBase. [130]

Experimental Part

4 Monitoring a waste-to-energy plant

4.1 Aim of the work

The aim of the experimental work is to design a scheme of a process monitoring system for a waste to energy plant case study based on the state-of-the-art artificial intelligence methods and software tools. Process monitoring plays important roles in industrial operations such as ensuring process safety, product quality, profitability as well as environmental management.

In the case study at hand, different process phenomena are critical in the overall performance of the process and thus require real-time or offline monitoring. For instance, quantification of the heating value of the feed waste material is important. However, due to the heterogeneous nature of feed waste material, measurement and monitoring its heating value can be difficult and challenging. There is also a restriction on the temperature inside the gas combustion chamber with a minimum value of 850°C required and it must be hold for at least two seconds, according to the 'EU Waste Burning Directive'. Moreover, flue gas generated is rich in dust particulates and condensable components which together lead to fouling of the heat exchanger surfaces inside the boiler. Mitigation of the adverse effects of fouling, necessitates application of soot blowers which in turn adds to process operational and economical burdens. Furthermore, it is also important to maintain constant fluidization of the gasification chamber because it is key to the overall performance of the plant.

Considering the above mentioned process phenomena, this work attempts to develop a monitoring system employing latest AI methods and software tools for real-time process monitoring. However, before developing the monitoring scheme, performances of some of the commonly used data-based process monitoring methods were examined. This was done by studying the implementation of these methods in solving key selected process phenomena related tasks. In the first task studied, the goal was to predict the temperature of flue gas at the exit of the combustion chamber. The second task focused on prediction of the syngas heating value based on the available data. The next task concentrated on examining the sootblowing intervals necessary in alleviation of fouling of heat exchanger surfaces as well as investigating its relationship with power generation efficiency since the sootblowing exercise utilizes part of the live steam. The final task involved understanding the influence of hot flue gas exit temperature and ammonia flow on NOx emissions as well as analyzing the dependence of hot flue gas and fluidized bed temperature on electrical power output.

4.2 Process description

In the process, solid waste is fed into the Outotec Advanced Staged Gasifier where it is combusted to produce heat which is recovered and used for electric power generation. The scheme in Figure 34 [131] summarizes the different stages of the waste-to-energy plant. The main process unit is the Outotec Advanced Staged Gasifier, which is divided into two parts, the lower and upper sections. In the lower section, gasification takes place in a bubbling fluidized bed where a sand bed and solid waste particles or fuel are fluidized by air. The gasification stage produces syngas which escapes to the upper chamber of the gasifier. At this point, any metallic components and large-sized particulates

are removed by discharging part of the bed material through the bottom cone of the fluidized bed. After cleaning, the bed material is recycled back to the gasifier as demonstrated in the process scheme (Figure 34). Combustion takes place in the upper section (also referred to as the vapor space) of the gasifier where syngas is completely combusted in presence of air. Ammonia is added in the upper chamber in order to reduce NO_x content in the flue gas. It reacts with NO_x compounds to form harmless nitrogen and water. The flue gas exits the gasifier at a temperature of 930ºC. Also the walls of the vapor space are fitted with heat exchange surfaces for heat recovery in the steam generation process.

Hot flue gas from the gasifier flows through a natural circulation boiler where saturated steam is further heated to produce super-heated steam. Subsequently, superheated steam is used to run steam turbines for electric power generation. After the boiler, flue gas is passed through multi-clone section for dust and gas separation. In the next stage, flue gas is sent to the economizer where heat is recovered by pre-heating of water feed to the boiler. Before discharge to the environment, flue gas goes through a scrubbing stage where pollutants such as sulfur dioxide are removed by neutralization and subsequent removal of any solid particulates with the help of filter bags. [131]

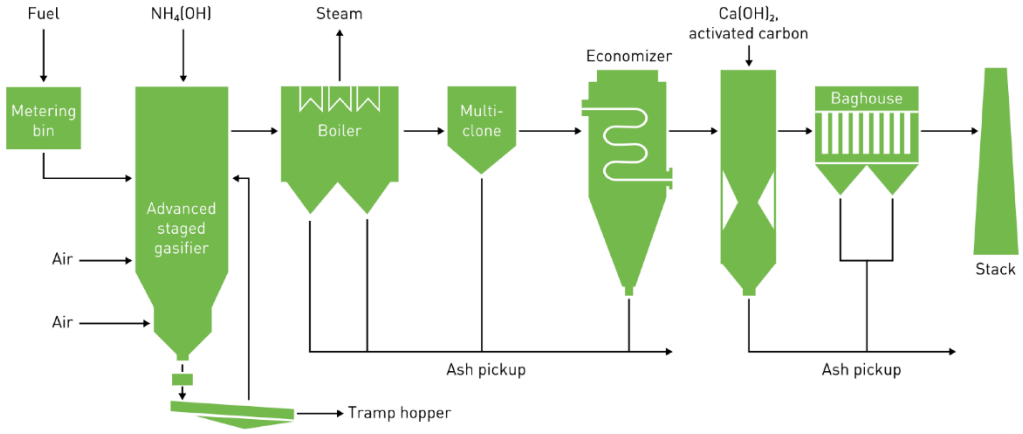

Figure 34. Schematic representation of Outotec® waste to energy plant. [131]

4.3 Process monitoring system description

A data-based process monitoring system is considered in this work. The monitoring scheme follows the general procedures which comprises of data acquisition, data-preprocessing, model design and model maintenance as shown in Figure 35. In this proposed process monitoring system another key aspect is the utilization of state of the art AI software tools and cloud computing technologies. The monitoring system also is expected to support both online real-time process monitoring as well as offline historical data analysis. Moreover, the monitoring system proposes leveraging big data distributed computing tools such as Apache spark, Hadoop among others to maximize fast data processing and analysis for real-time process monitoring.

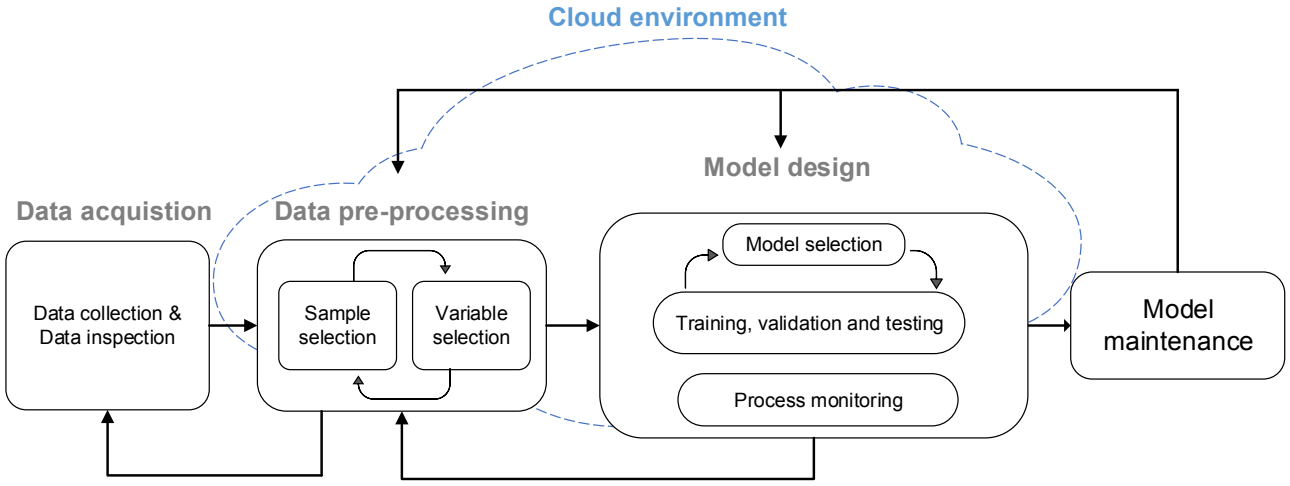

Figure 35. A simplified scheme for process monitoring.

4.3.1 Data acquisition

As demonstrated in Figure 35, data acquisition (DAQ) is considered here as the first step of a process monitoring system. The DAQ layer enables measurements from different local devices in the process to be collected for purposes which may include data inspection or visualization, analysis, storage and so forth. In industrial processes data acquisition can be accomplished in a number of interfaces for instance use of OPC UA, OPC, Modbus, various network protocols and so on as illustrated in Table 1. Moreover, in Chapter 2.3 we saw that currently various industrial IoT platforms can be implemented for data collection from industrial devices. According to Table 1, the architecture of a data acquisition system influences the functionalities of a process monitoring system. For example in case of a real-time process monitoring requirement, industrial interfaces like OLE process Control (OPC), Modbus and MT Connect are necessary. In case of smart devices, network protocols such as MQTT, COAP and HTTP also support real-time monitoring. Otherwise, in some cases due to high latency behavior only offline or batch data analysis can be achieved

Table 1. Typical sources of industrial equipment data. [132]

Classification	Data source	Data type	Common interfaces	End user	Latency
Database	Building management system (BMS)	Energy and environmental	ODBC,OLEDB, JDBC etc.	Energy and facilities	Batch
Database	Monitoring and targeting (M&T)	Energy	ODBC,OLEDB, JDBC etc.	Energy and facilities	Batch
Database	Manufacturing execution system (MES)	Production and automation	ODBC,OLEDB, JDBC etc.	Automation, production and quality	Real-time and batch
Device	Programmable logic controllers (PLC)	Production, energy and environmental	OPC, MT Connect, BACnet, Modbus, LonWorks	Automation and building services	Real-time and batch
Device	Gateways	Multiple	HTTP,OPC, Modbus, I/O (i.e. CSV)	Multiple	Real-time and batch
Device	Smart devices (i.e. IoT)	Multiple	MQTT, COAP, HTTP	Multiple	Real-time

4.3.2 Data pre-processing

In general, raw-data is often obstructed with anomaly features which can be attributed to noise, disturbances, uncertainties, measurement errors and so forth. For instance, process data can be affected by intermittent plant shut downs (either planned or not), equipment failures, power surges, random measurement errors (related to equipment or due to personnel), test work campaigns among other things. Moreover, in some cases datasets tend to vary widely in measurement scales which makes data analysis difficult. This implies that direct use of data often leads to poor analytical results. Therefore, it is important to subject raw-data to data pre-processing procedures before the data can be used for analysis. Data pre-processing methods are meant to remove features like noise, outliers, missing data, duplicate data, irrelevant data (in case of process shutdowns, equipment malfunctions, etc.), varying data scales and so forth. In other words, data pre-processing methods improve the quality of data thereby enhancing the accuracy of analytical results after data analysis. [133]

The most common steps applied in data-preprocessing are data visualization, selection of relevant datasets, outlier removal, scaling and data filtering. Other additional data pre-processing measures may include resampling, delay estimation and compensation, data transformations and data reduction. [133]

In many cases, the first thing to do is data visualization to assess the quality of data at hand. This step also provides insights on how to proceed with data pre-processing such as selection of most relevant parts of the data. For example, with data visualizations missing values, plant shutdowns and any other easily visible data inconsistencies can be identified. Depending on the application at hand, certain parts of the data are often excluded for instance data corresponding to plant shutdowns, equipment failure and so on. Inconsistent data points or outliers also can be manually removed through visual inspection and use of 'thresholds' based methods. A zero-order hold among other criteria is often applied when dealing with removal of inconsistent data or missing values. Outliers are normally classified into two categories global and local types where the former represents observations which are inconsistent with the overall dataset and the latter focuses on data inconsistencies within a localized neighborhood. The presence of outliers can be due to measurement errors, faulty devices, data contamination and so forth. The removal of outliers can be done in several ways, for instance by use of a 'three-sigma threshold' technique [133]. The Hampel identifier method [134] is one similar technique that is often used to detect and removal outliers. Scaling of data can be done either using the standardization method or by the normalization technique. In the standardization method, the idea is to obtain a normally distributed data having a zero mean and a unit variance whereas in the latter method, data is scaled to a range usually of zero to unit. It is also important to note that data scaling is necessary before applying most of the machine learning algorithms such as PCA (principle component analysis), PLS (partial least squares), artificial neural networks, among others. The other common data pre-processing step is filtering which is done to remove or minimize noise contained in the data. Methods which can be applied for this purpose include a moving medium filter (MMF) and a moving average filter among others. [133]

4.3.3 Model design

Model design involves tasks like model selection, model training, model validation and model testing. In model selection an appropriate method is identified and the algorithm is developed if necessary or a ready-made algorithm is utilized. For this case study, some machine learning and deep learning software tools discussed in Chapters 2 are proposed. The choice of an appropriate method is influenced by the process behavior such as time dependent systems, dynamic systems, data dimensionalities, non-linear or linear systems and so forth. For instance, dynamic principal-component analysis (DPCA), recursive principal-component analysis (RPCA) and moving window principal component analysis (MWPCA) are applicable in cases of time-dependent and high dimensional data [135]. Furthermore, process monitoring environments such as real-time and offline data analysis also influence the best suitable monitoring method applicable. Examples of data-based monitoring methods and their potential areas of application are presented in Figure 36.

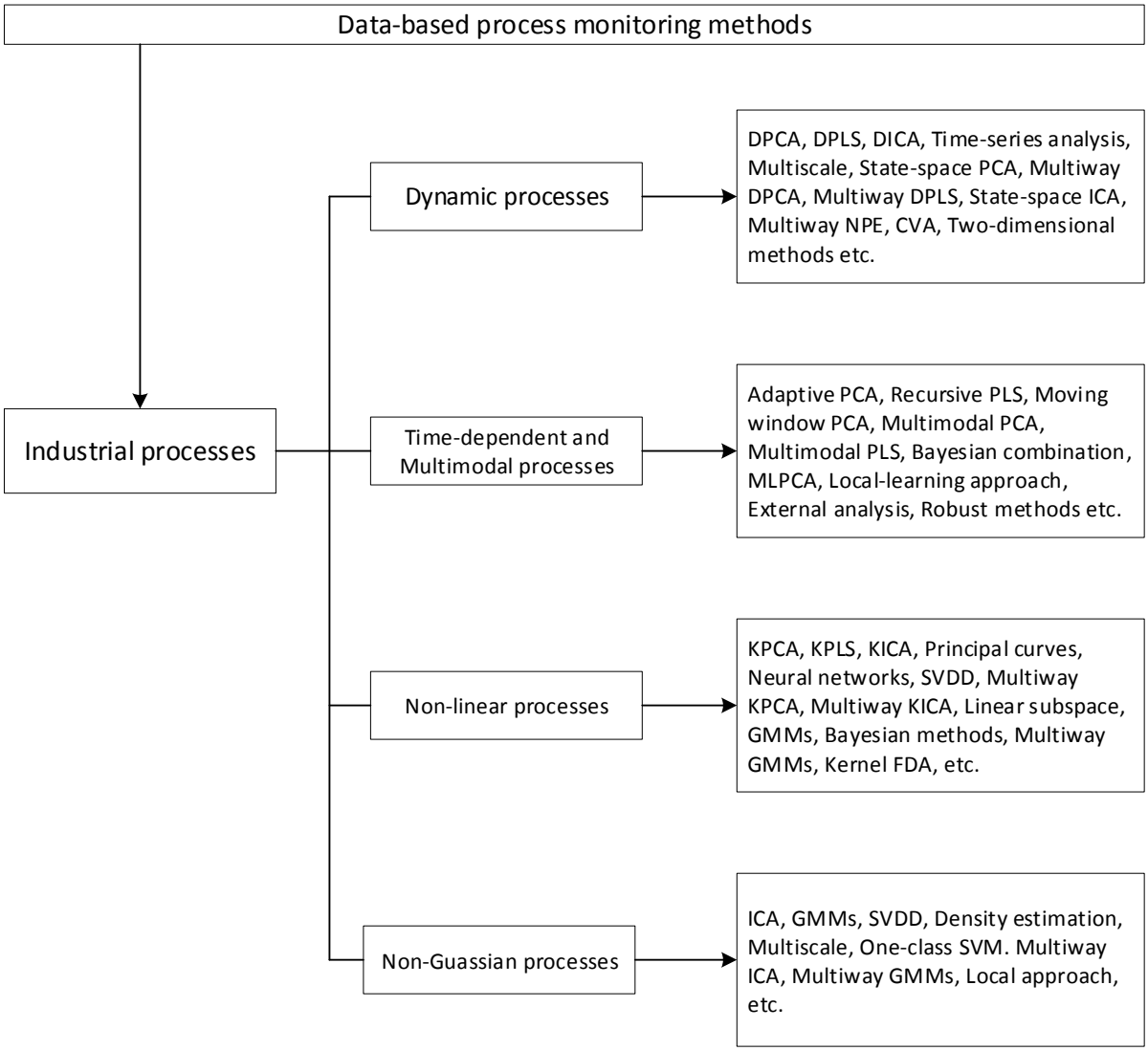

Figure 36. Different data-based monitoring methods. [135]

4.3.4 Model Maintenance

Although the model created can be used without further changes, sometimes due to process behavior such as sensor drifting and changes in process data, a model may become obsolete. Therefore it is important to periodically evaluate the performance of the model and where necessary make modifications. Some of the methods which are often implemented in model maintenance are similar to those employed in time-varying and multimodal processes for example, use of recursive or adaptive modelling methods, moving-window techniques and multimodal methods.

4.4 Implementation of data-based process monitoring methods

Previously, a group of researchers proposed and tested different statistical methods in solving some of the typical process phenomena associated with this particular waste-to-energy plant, for which process data was available. These same tasks were briefly introduced in Chapter 4.2. The statistical methods employed here are among the traditional and commonly used process monitoring methods. These included principal component analysis, partial least squares, multivariable linear regression, general linear regression, artificial neural network, among others. A number of software tools (testing environments) were also utilized notably, Kasem, Matlab, RStudio and Scikit-learn. The methods, testing environments applied and the results obtained in each respective task, are presented in this Chapter 4.4.

4.4.1 Prediction of combustion chamber exit temperature

This particular task was investigated by a number of researchers with use of different methods and software tools. Notable individual works are described here.

Methods and testing environment

In the first instance [134] a multivariable linear regression method was proposed for the development of a prediction model for flue gas temperature at the exit of the combustion chamber (T4). Figure 37, shows the 'Outotec Advanced Staged Gasifier', other process units as well as the key streams and variables studied. In data pre-processing, outliers were removed by use of the Hampel identifier technique. Thereafter, the data was filtered using the moving average filter method. In order to select the variables affecting the prediction of the exit temperature of the combustion chamber, a correlation matrix was developed for variables in Table 2. From the correlation analysis variables T1, T2, T3, P1, P2, V1, V2, q2, q8, q9 were found to be correlated to the flue gas temperature, T4. (See Appendix A, Table A-1). With the exception of ammonia flow, q9, these variables were used to develop the prediction model for flue gas exit temperature, T4 using a multivariable linear regression method. It is important to note that the variable q9 was excluded from the model prediction variables because ammonia is only intermittently added to the combustion chamber in response to changes in flue gas temperature with an ultimate purpose of controlling NOx emissions in the flue gas. The delay between selected predictor variables was estimated based on the cross-correlation technique. The multivariable linear regression method demonstrated in Eqn. (1) was applied, where Y is a vector of prediction variables, X is a matrix of predictor variables and β is a vector of estimated parameters or model coefficients and ε, error (noise) estimation.

$$Y = \beta X + \varepsilon \tag{1}$$

where, $Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$, $X = \begin{bmatrix} 1 & x_{11} & \cdots & x_{1p} \\ 1 & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n1} & \cdots & x_{np} \end{bmatrix}$ and $\beta = \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \left(X^T X\right)^{-1} X^T Y$

The methods above were implemented in MATLAB® 2017 software, this included data-processing and estimation of the parameters of the model shown in Eqn. (2). It is important to note here that, the non-linear terms of the model in Eqn. (2) were determined using a trial & error approach after inspecting the trends of the given process data.

$$T_4 = a_0 + a_1 T_1 + a_2 T_2 + a_3 P_2 + a_4 lnV_1 + a_5 lnV_2 + a_6 lnq_1 + a_7 lnq_8 + a_8 \sqrt{q_2} \tag{2}$$

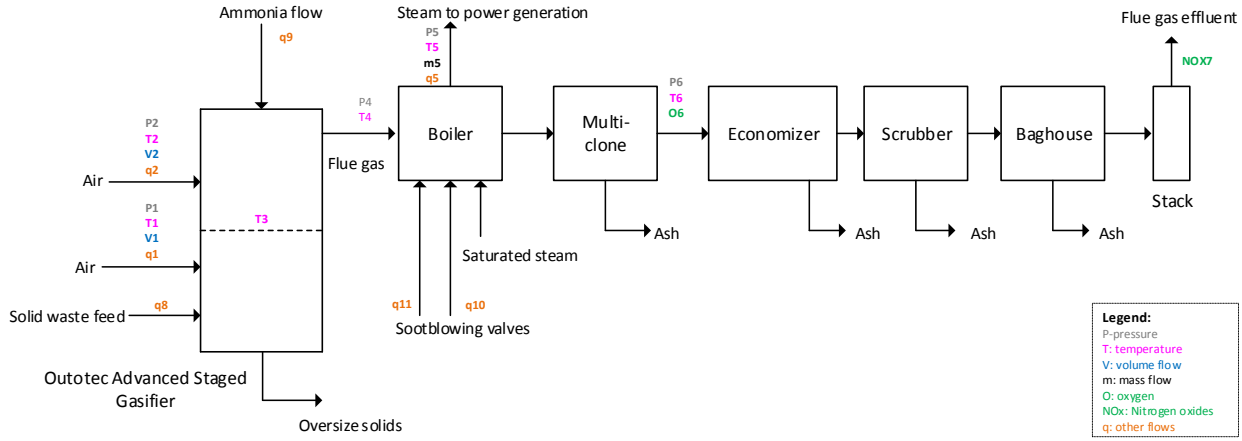

Figure 37. A simplified process scheme showing relevant streams and variables.

Table 2. List of variables and their descriptions.

Variable name	Variable description
T1	Air temperature to the fluidized bed
T2	Air temperature to the combustion chamber
T3	Fluidized bed temperature
T4	Temperature of flue gas to the boiler
T5	Temperature of steam from the boiler
T6	Temperature of flue gas to the economizer
P1	Air pressure to the fluidized bed
P2	Air pressure to the combustion chamber
P4	Pressure of flue gas to the boiler
P5	Pressure of steam from the boiler
P6	Pressure of flue gas to the economizer
V1	Volumetric flow of air to the fluidized bed
V2	Volumetric flow of air to the combustion chamber
m5	Mass flow of steam from the boiler
q1	Air valve opening to the fluidized bed
q2	Air valve opening to the combustion chamber
q5	Steam valve opening to power generation
q8	Solid waste feed flow
q9	Ammonia flow
q10	Sootblowing valve 1 opening to boiler
q11	Sootblowing valve 2 opening to boiler
O6	Oxygen content in flue gas to the economizer
NOx7	Nitrogen oxides content in flue gas effluent
EP	Electrical power output

Schach and Kupka [136] also developed a model for the prediction of flue gas exit temperature, T4 using three different statistical methods. Data pre-processing and analysis was done using RStudio (an open source software written in R programming language). After data pre-processing, which included smoothing of signals, outlier removal, correlation estimation and scaling, variables P2, q2, q8 and q9 were selected for further studies. Unlike in the previous work, [134] variable T3 was found to have no correlation with T4 and thus eliminated in from the predictive model. Moreover, with the implementation of principle component analysis (PCA) technique they were able to reduce the number of dimensions of the selected dataset of variables. With two principle components considered, a prediction model was developed using the principle component regression (PCR) method. The researchers also modeled the temperature, T4 by employing a linear model after data transformation with the box-cox technique. The box-cox-transformation was realized with use of the 'box-coxfit' function of the 'geoR package' library. Temperature prediction was also done using the artificial neural network (ANN) method with two hidden layers. The 'neuralnet' library in R was utilized for this purpose. The neural network scheme is presented in Figure 38.

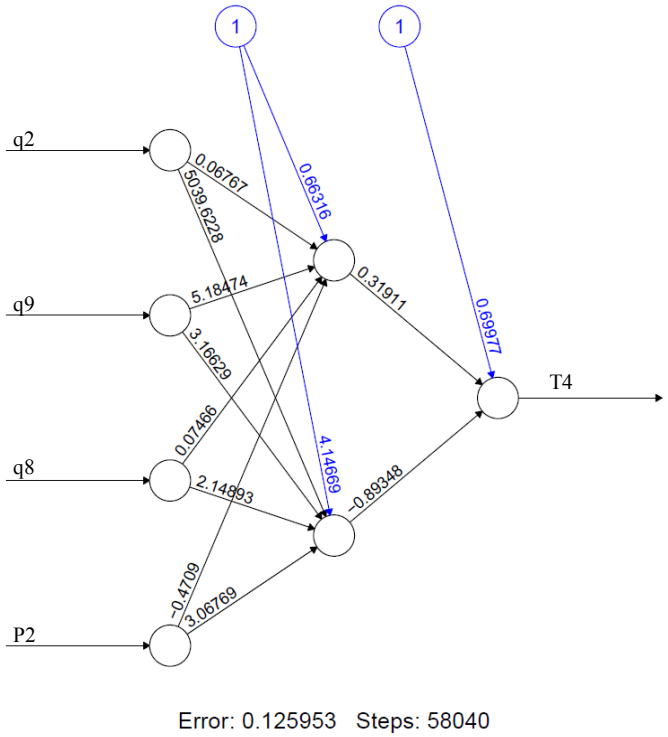

Figure 38. Neural network scheme with error prediction and number of iterations. [136]

The same task was attempted by several other researchers, for instance Burchell [137] applied a partial least square (PLS) regression method to develop a model for prediction of flue gas temperature at the exit of the Outotec Advanced Staged Gasifier. All variables listed in Table 2 were examined while developing the predictive model. The modeling work was performed using Scikit-learn, which is a machine learning library available under an open source license. Other software tools used included numpy (for linear algebraic tools), pandas (for data handling) and Matlab tools (in implementing the 'variable importance in projection', VIP technique after generating the PLS model).

In another similar work, Silva [138] applied PLS regression and principal component regression (PCR) techniques. Here, modeling work was done using Matlab software tools for PLS and PCA. Kasem software was employed in data pre-preprocessing which involved data filtering and correlation studies. In data filtering, an 'aberrant filter' and a 'median filter' were applied in the respective order. Correlation studies were carried out using Pearson (linear) and Spearman (non-linear) methods. However, selection of key variables for building the prediction model was done based on correlation results from Pearson's linear method.

Furthermore, Nuorivaara [139] implemented a 'multiple iterative regression' method in Kasem software to generate models that are able to predict the temperature of hot flue gas leaving the gasifier. Prior to model building, process data was filtered using 'aberrant values filter' and after a 'median filter', available in the Kasem software. In this case, two predictive models were generated in a form of a polynomial with respect to each variable as demonstrated in Eqn.s (3) & (4). The difference in the two models is the replacement of variable T3 (fluidized bed temperature) in Eqn. (3) with P2 (pressure of air to the combustion chamber) in Eqn. (4).

$$T_4 = a_0 + a_1T_2 + a_2T_2^2 + a_3T_3 + a_4T_3^2 + a_5V_2 + a_6q_2 + a_7q_8 + a_8q_8^2 \quad (3)$$

$$T_4 = a_0 + a_1T_2 + a_2T_2^2 + a_3P_2 + a_4P_2^2 + a_5V_2 + a_6q_2 + a_7q_8 + a_8q_8^2 \quad (4)$$

Where, T_4 is the predicted flue gas temperature, T_2 is the temperature of air to the combustion chamber, T_3 is the temperature of fluidized bed, V_2 is the volumetric flow of air to the combustion chamber, q_2 is the valve opening of air to combustion chamber, q_8 is the solid waste feed, P_2 is the pressure of air to the combustion chamber and $a_0, a_1, a_2, a_3, \ldots, a_8$ are model parameters.

Similarly to the proceeding work, Ai [140] also used a 'multiple iterative regression' method in Kasem software to build a model for predicting the temperature of hot flue gas exiting the combustion chamber. Again, Kasem tools for data pre-processing were applied in visual inspection, filtering and correlation estimations. The variables studied are the same as in Table 2. However, a list of variables which included V1, P2, q1, q8, q9 in addition to T4 were considered in model development after correlation analysis. Other than the multiple iterative regression technique, the researcher also applied the artificial neural network (ANN) technique to predict T4 using the same variables, V1, P2, q1, q8 and q9. The neural network model was implemented using Matlab software. Other researchers, Wang [141] and Penkov [142] also studied the use of a multiple iterative regression method in Kasem software for generating a predictive model for T4.

Results and discussions

The performance of the multivariable linear regression (MLR) model described by Eqn. (2) is demonstrated in Figure 39. Model validation was done using 30% of the process data with the rest of the data used for model training. The estimated prediction accuracy of the model was about 82%. The coefficients of the model Eqn. (2) are provided in the Appendix (Appendix A, Table A-1).

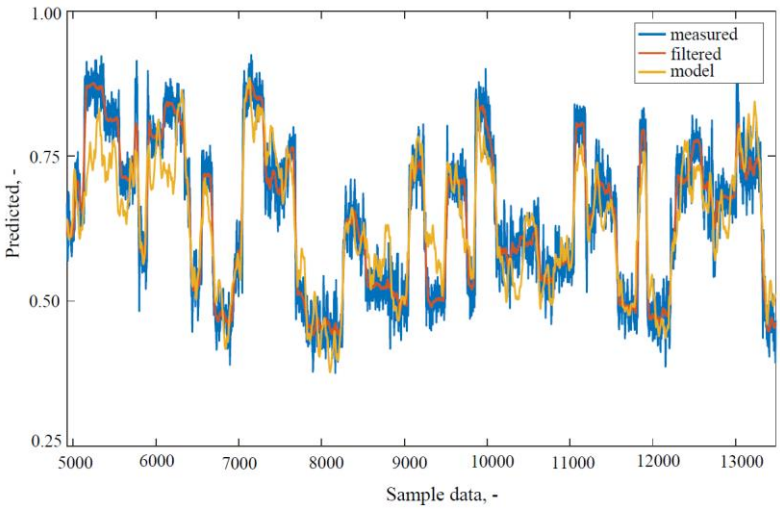

Figure 39. Performance of a MLR model in predicting flue gas temperature, T4. [134]

The results from the principal component regression (PCR) model [136] are summarized in Figure 40. According to Figure 40, the behavior of flue gas temperature, T4 was described fairly well by a two principal component model. The model prediction error did not significantly decrease in the case of more than two principal components. In addition, it can be observed that the measured process data shows a non-linear behavior which could not be captured quite well by the regression model.

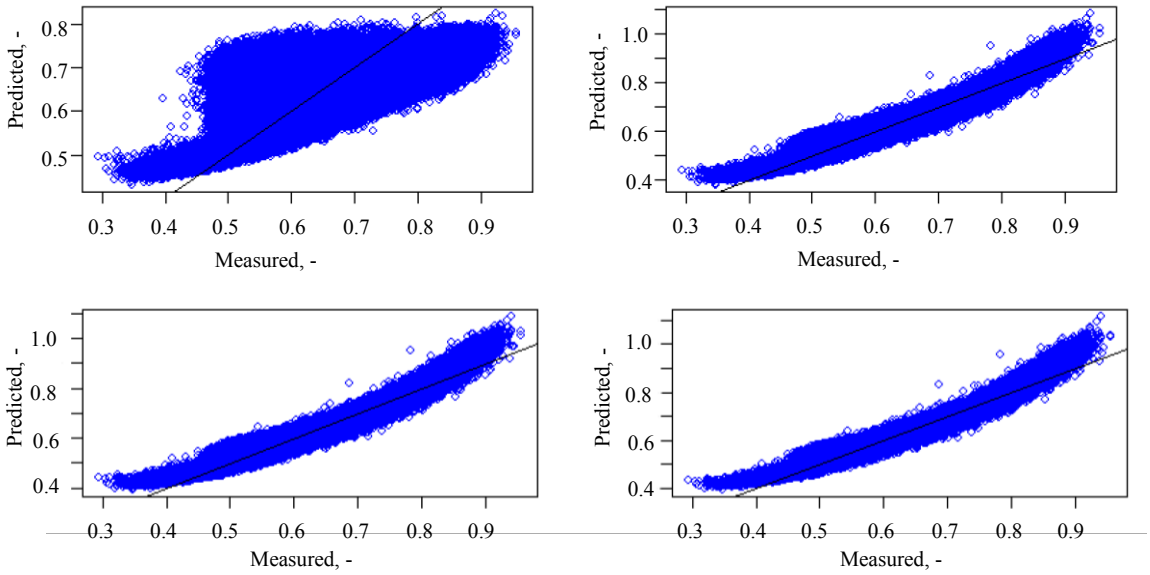

Figure 40. Performance of the PCR model in predicting variable T4: using, one principal component (top left), two principal components (top right); three principal components (bottom left) and four principal components (bottom right). [136]

To address the non-linear behavior, the researchers [136] developed a neural network based prediction model and its performance is illustrated in Figure 41. Although the neural network model showed better prediction accuracy than the PCR model based on calculated mean absolute percentage error, it was noted to be slow in converging to a solution. This implies that it would be difficult to apply an ANN based model in real-time process monitoring. Nevertheless, such a model can be used for offline prediction of flue gas temperature, T4. Furthermore, again in Figure 41, the researchers [136] presented a 'simpler' linear model based on 'box-cox transformation' of process data. The linear model has four parameters (See Appendix A, Table A-2) with variables, q2, q8 and q9. The linear model displayed better performance as compared to the PCR model according to prediction errors of 0.26% and 0.48% respectively. Moreover, convergence of the linear model was considerably fast. Thus, it is suitable for real-time temperature prediction.

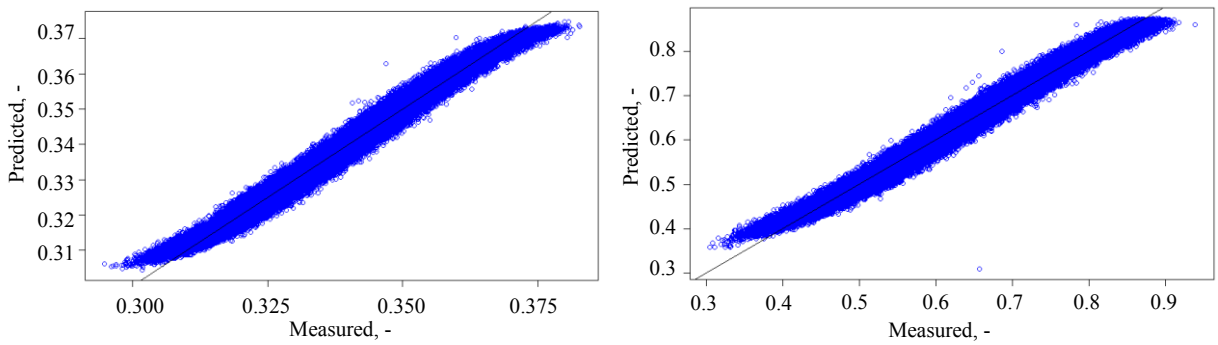

Figure 41. Prediction plots for flue gas temperature, T4: artificial neural network model (left) and the linear model with transformed variables (right). [136]

The results of a PLS regression model [137] are presented in Figure 42. After a 10-fold cross-validation, a PLS model with four factors was developed. As seen from Figure 42, the PLS model predicted flue gas temperature (T4) exiting the combustion chamber with quite good prediction accuracy based on R^2 of 0.95 reported on a test dataset. However, the data showed a non-linear behavior (See Appendix B, Figure B-1) which the PLS regression model could not describe, similarly to previous cases [136]. After examining the model's weights, regression coefficients and the 'variable importance in prediction (VIP)', the observed key variables for flue gas prediction were ammonia flow (q9), pressure of air to the combustion chamber (P2) and the feed rate of fuel material to the gasifier (q8).

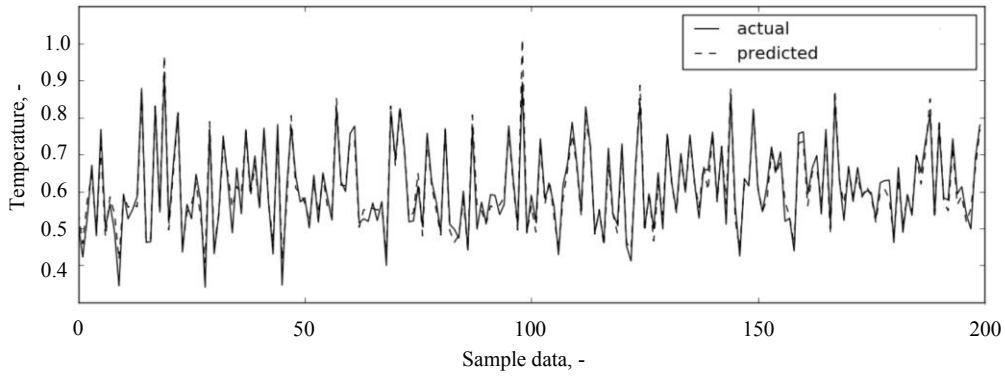

Figure 42. Performance of PLS regression model in predicting flue gas temperature, T4. [137]

From Silva's work [138] results of PLS regression model and PCR models are presented in Figure 43, where three respective components were used in each case. The performance of the two models was comparably similar. However, as with previous researchers, the models would be limited in a certain range of temperature, T4 due to a non-linearity behaviour as it can observed in Figure 43. The key predictive variables chosen here were ammonia flow (q9) and pressure of air to the combustion chamber (P2).

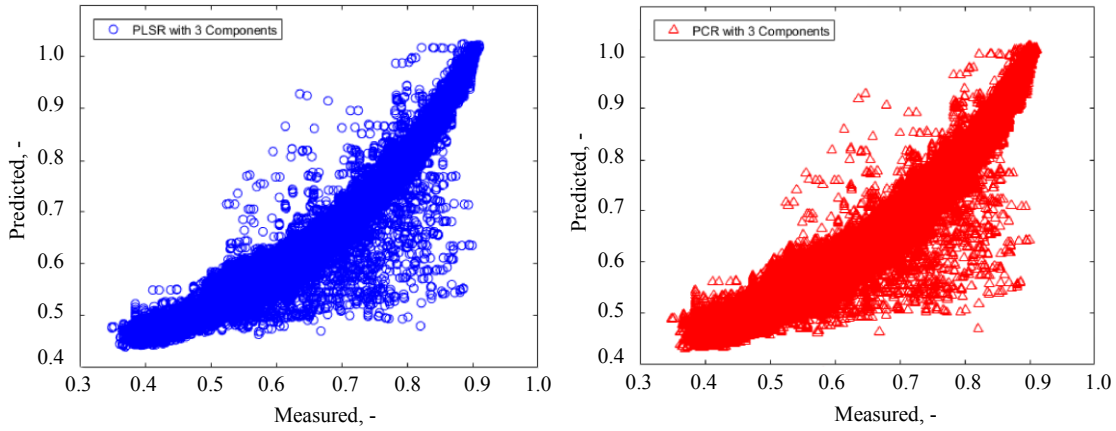

Figure 43. Predictive PLS regression model (left) and PCR model (right) for variable T4. [138]

The performance of multiple iterative regression models developed by Nuorivaara [139] is illustrated in Figures 44& 45. It can be seen that the two models predicted the measured flue gas temperature, T4 fairly well. Notable performance difference between the two predictive models can be seen in the spikes and comparison of the model residuals. The first model in Figure 44 showed less distorted peaks as compared to the second model in Figure 45. However, the former model appeared to be noisier than the latter according to the respective residual histograms.

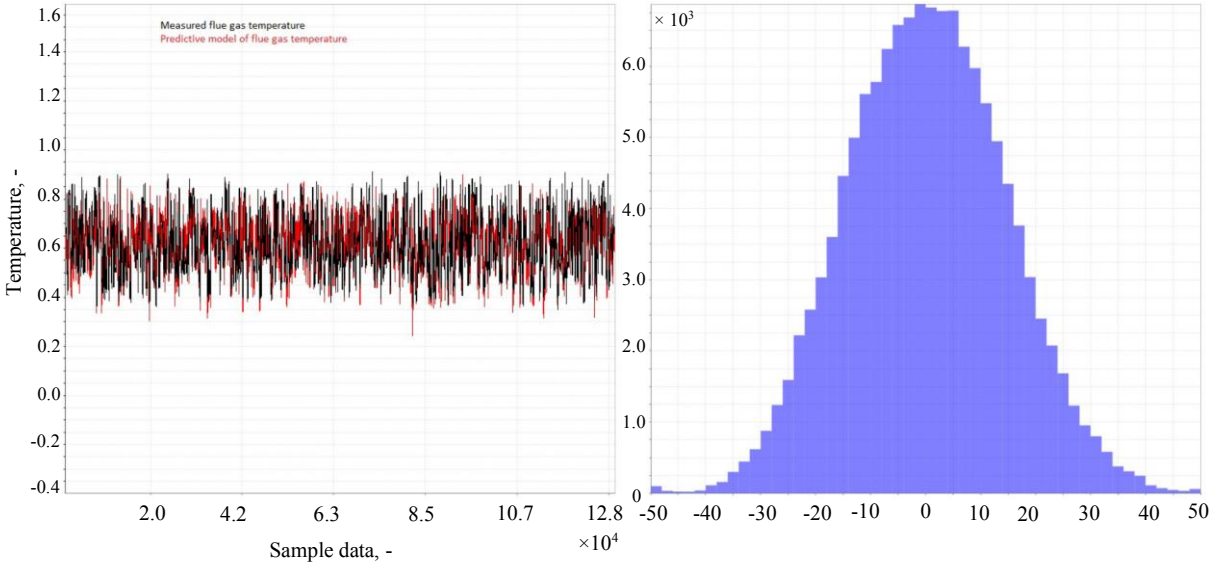

Figure 44. Predictive model for flue gas temperature, T4 described by Eqn. (3). [139]

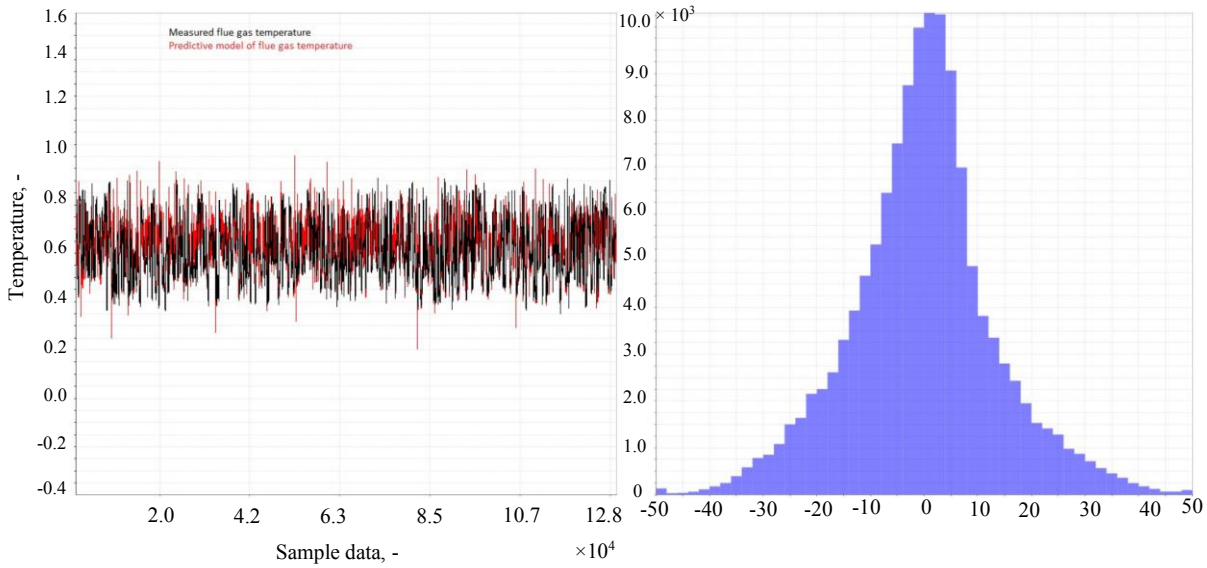

Figure 45. Predictive model for flue gas temperature, T4 described by Eqn. (4). [139]

Again, the neural network [140] based model performed quite well in predicting the flue gas temperature, T4. The performance of the neural network based model is shown in Figure 46. On the other hand, a predictive model based on multiple iterative regression [140] also performed fairly well in describing the flue gas temperature (See Appendix B, Figure B-2). The key difference between the two models, lie in the ability to capture any non-linear behaviour in the process data utilized. This was observed by inspection of the residuals distribution. With the neural network based model, a normal distribution of the residuals was observed (Figure 46) unlike in the case of a multiple iterative regression model.

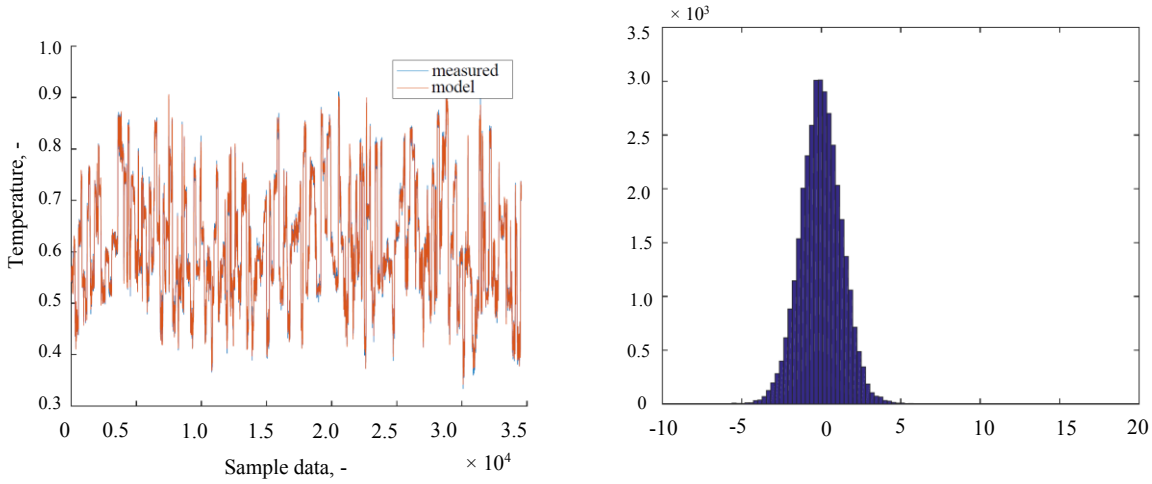

Figure 46. Predictive model based on a neural network: left, model prediction and right, residual distribution. [140]

To summarize the findings from this task, models developed by many researchers could only describe the process behaviour in a limited process operation range. This was mainly due to inherent non-linearity relationships between the selected variables in each particular case. Moreover, the lack of sufficient process knowledge affected the judgement in the selection of key variables. For instance, most of the researchers included ammonia flow, q9 in the prediction model for exit temperature, T4. Based on the process operations, ammonia is intermittently added after an increase in T4 is observed and hence no dependency of T4 on ammonia addition is expected. Therefore, predictive models developed mainly based on variable q9 do not fully capture this particular process phenomenon. Another notable item is the better performance of artificial neural network models as compared to other studied methods. Neural Network based models are at the centre of the current wave of AI as mentioned earlier in Chapters 2. They are able to handle complex non-linear systems. Although, convergence of ANN models was reportedly [136] slow, with the state-of-the art computing tools, faster computation times can be achieved.

4.4.2 Prediction of syngas heating value

Patscheider and Weingrill [143] developed a physical model for prediction of syngas heating value based on process variables, T1, T2, T3, T4, P1, P2, P4, V1, V2 and O6 (See Figure 37 & Table 2).

Methods and testing environment

In this case, data pre-processing was implemented in the Kasem software program, first by visual inspection of raw-data plots and selection of relevant datasets. The second step involved data filtering using 'aberrant values filter' and a median filter for data 'spikes' removal and 'smoothing' data trends respectively. In addition to data-preprocessing, the heat capacity of the flue gas at the exit of the combustion chamber was determined with use of HSC Chemistry Software. It was also assumed that the major components of flue gas were N_2, O_2 and CO_2 (wt.-%). Nitrogen content in the flue gas was assumed to be 75 wt.-%. Knowing O_2 content, the amount of CO_2 in the flue gas was determined accordingly from the gas component balance. The formula in Eqn. (5) was used to calculate the heat capacity of the flue gas (cp4) at temperature, T4 as well as heat capacities for air to the fluidized bed (cp1) at ambient conditions and for syngas (cp3) at T3. The composition of syngas was assumed to be 75 wt.-% N_2, 12.5 wt.-% CO and 12.5 wt.-% CO_2.

$$c_p = A + B\frac{T}{10^3} + C\frac{10^5}{T^2} + D\frac{T^2}{10^6} \tag{5}$$

where, c_p is the specific heat capacity (kJ/kgK), T is temperature (K), A, B, C and D are constants.

After determination of the relevant heat capacities of gaseous streams, the respective enthalpies of the streams were calculated as demonstrated in Eqn. (6). From the heat balance, the enthalpy due to the oxidation reaction, ΔQ (kJ) was estimated according to Eqn. (7). To obtain the heating value of syngas, the enthalpy ΔQ was divided by the mass flow of air to the fluidized bed. Here, the mass flow of air was obtained using its volumetric flow, V1 and equivalent density.

$$Q_4 = m_4 c_{p4} T_4 \quad (6)$$

$$\Delta Q = Q_4 - (Q_1 - Q_3) \quad (7)$$

where, Q_4 is the enthalpy of flue gas (kJ), m_4 is the mass flow of flue gas (kg/s), c_{p4} is the specific heat capacity of flue gas (kJ/kgK), T_4 is the flue gas temperature (K), Q_1 is the enthalpy of air to the fluidized bed (kJ) and Q_3 is the enthalpy of syngas (kJ).

Results and Discussions

The behaviour of heating value is presented in the Figure 47. It can be observed that the heating value of syngas (on a scale of unit) fluctuates in between 0.20 to 0.88. The observed behaviour is partly attributed to the varying nature of solid fuel to the gasifier [143]. Moreover, the results obtained were within the range of expected values according to other plant data studies. [144]

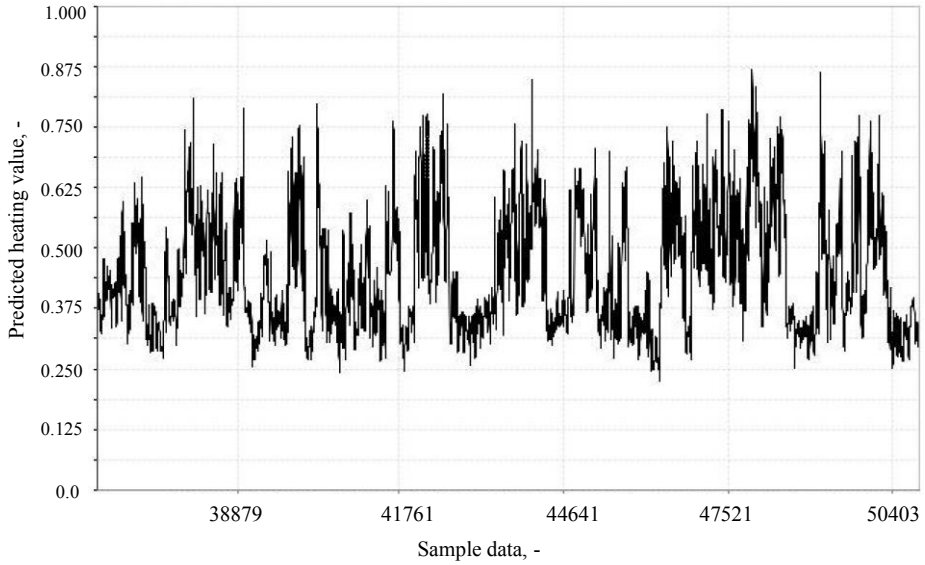

Figure 47. Calculated heating value of syngas. [143]

It is important to note here that, measured syngas heating values were not available during model development and thus the two datasets could not be directly compared in order to estimate the performance of the model. Moreover, the physical model did not take into account any heat losses around the affected process units because such information was unavailable. This task differs from the preceding task in the sense that the results are solely based on a physical (heat balance) model rather than a data-based model. The variables used in the actual predictive model are calculated variables and subject to the assumptions made prior to modeling as described in the methodology.

4.4.3 Examining the soot blowing interval and electrical power output

According to the problem statement, the process data studied represented different soot-blowing conditions. For example, under test campaigns and during normal plant operations, soot-blowing intervals employed varied. Researchers were tasked to study the most effective soot-blowing interval as well as its relationship with electrical power output. Table 3 presents the sootblowing intervals for each studied period of plant operation.

Methods and testing environment

Schach and Kupka [136] tackled this problem by first estimating the representative soot-blowing intervals for the two distinct periods, i.e. period 6 and period 7. They reported respective soot-blowing intervals of 0.71 and 0.59. Details on how they arrived at those values were not clearly explained in their work. Nevertheless, relevant variables affecting electrical power output of the plant (EP) were determined through correlation analysis. The selected variables included oxygen content in flue gas (O6), steam temperature (T5), steam valve opening (q5), ammonia flow (q9) and flue gas pressure (P6). It is important to note here that, power output and sootblowing interval did not show considerable correlation as presumably expected. However, further comparison of the mean values of variables showed a dependency of power output on sootblowing interval. According the researchers [136], such dependency cannot be described by only an individual variable. Thus, in order to understand the fouling problem, 'ratios and relative differences' of variables in relation to sootblowing interval were computed and used in the modeling of electrical power output. The modeling work was carried out in RStudio with the use of statistical methods such as PCA (for dimension reduction) and PCR for model parameter estimation.

Table 3. Sootblowing intervals representing the studied process data. [131]

Operational period	**Sootblowing Interval (no units)**
1	0.08
2	0.13
3	0.17
4	0.33
5	0.67
6	long
7	short

Note. Scale is 0-1 for shortest and longest interval respectively.

Elsewhere, Catalina [145] used Microsoft excel tools to investigate the optimum sootblowing interval from the process data. Data-preprocessing and analysis both were implemented in Microsoft Excel 2007. In data pre-processing, selection of relevant variables was done after analyzing the mean and standard deviation of each variable. For instance, variables with zero mean values were immediately rejected from the variable list. Moreover, data was separated into several parts where each part corresponded to the respective sootblowing periods according to Table 3. Due to the limitations of Excel spreadsheets in statistical handling of data, only sections of the datasets were utilized. Also to capture more information from the data, in some instances either ratios or differences of variables were used instead of individual variable data.

In another work, Kieliba [146] investigated the same problem by using a 'one-way analysis of variance (ANOVA)' technique in addition to use of partial least square method. These methods were

implemented in Statistica software. In data pre-processing, notable measures taken included data cleaning using the 3-sigma threshold method, scaling and centering (prior to implementation of PLS analysis). The problem was approached in three steps, obtaining the optimum frequency of sootblowing using the 'one-way ANOVA' method, use of the PLS technique to observe correlation between other variables with sootblowing interval and lastly analyzing the fouling phenomena in relation to sootblowing intervals and electrical power output.

In additional similar works, Kossakowska [147] implemented different methods like general linear model (GLM), PLS and PCA in Statistica software. Also notable in this particular work, new variables which describe the fouling phenomena were created based on pressure and temperature differences. As with many previous studies, relevant variables were selected after correlation analysis studies. The variable list studied varied with the employed statistical method. For instance with the GLM technique, variables investigated included T4, T6, P4, P6, q10, q11, sootblowing interval and electrical power output (EP). In the case of PLS and PCA methods, more variables such as V2 and m5 were examined. Kubik [148] also applied PCA and PLS methods in the Kasem software to investigate the most suitable sootblowing interval as well as analyzing its relationship with electric power generation efficiency. The variables studied included m5, P5, q5, q10, q11, EP, and sootblowing interval.

Results and Discussions

Schach [136] incorporated the sootblowing intervals for respective periods in developing a PCR based predictive model for electrical power output and the results are presented in Figure 48. It can be seen that the performance of the model against measured process data was quite good. The key variables applied in the model are ammonia flow (q9), the observed relative pressure difference across the boiler (i.e. between P4 and P6) and sootblowing interval. Hence, the most economical sootblowing interval can be selected based on the desired electrical power output and alongside other model variables.

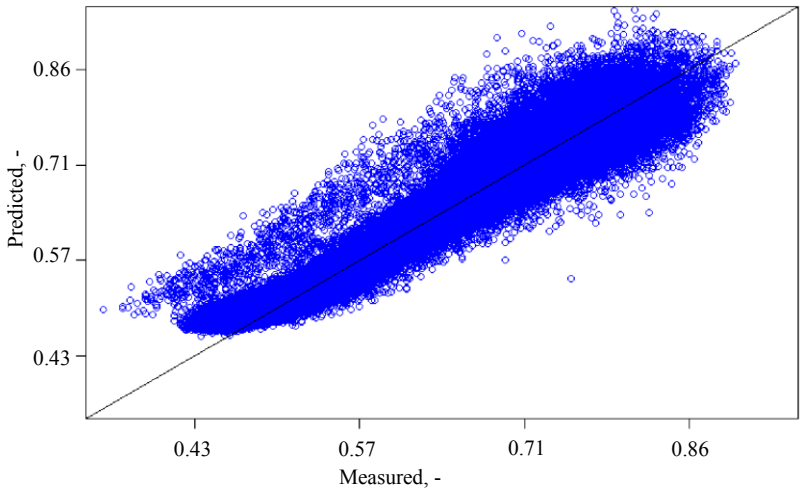

Figure 48. A PCR based model for electrical power output prediction. [136]

According to Catalina' work [145], sootblowing intervals equivalent to 0.33 and 0.67 offer better electrical power output as demonstrated in Table 4. However, the researcher also recognized the effect of sootingblowing interval on other variables such as temperatures and pressures around the boiler. Further studies (See Appendix A, Table A-3) revealed that changes in other process variables are quite significant in the case of long sootblowing interval of 0.67. This is partly attributed to fouling of heat exchanger surfaces. Therefore, the most economical sootblowing interval in this case was observed to be equivalent to the value of 0.33.

Table 4. Change in average electric power output after sootblowing. [145]

Sootblowing interval	**Power change after sootblowing, %**
0.08	-0.11
0.13	-0.66
0.17	-0.37
0.33	1.57
0.67	3.14

Furthermore, Kossakowska [147] observed that sootblowing intervals of 0.08 and 0.33 were key to higher electrical power output as demonstrated in Figure 49. It can be seen that sootblowing interval of 0.17 was the least effective in achieving high power output. In Figure 50, the performance of the general linear predictive models for electrical power output is displayed. The model represented as 'scenario I' was developed using the variable list composed of T4, T6, P4, P6, q10 and q11 in addition to the sootblowing Interval. The performance of this model in relation to plant data was quite good as seen from Figure 50 (left) with R^2 equivalent to 0.969 observed. In 'scenario II', the model was developed based on calculated variables describing the changes in heat and pressure. The results are presented in Figure 50 (right) where model prediction of measured data was equally good with R^2 of 0.923 noted. Although other methods such as PLS and PCA were applied by the same author, only results obtained with the general linear model are presented in this current work.

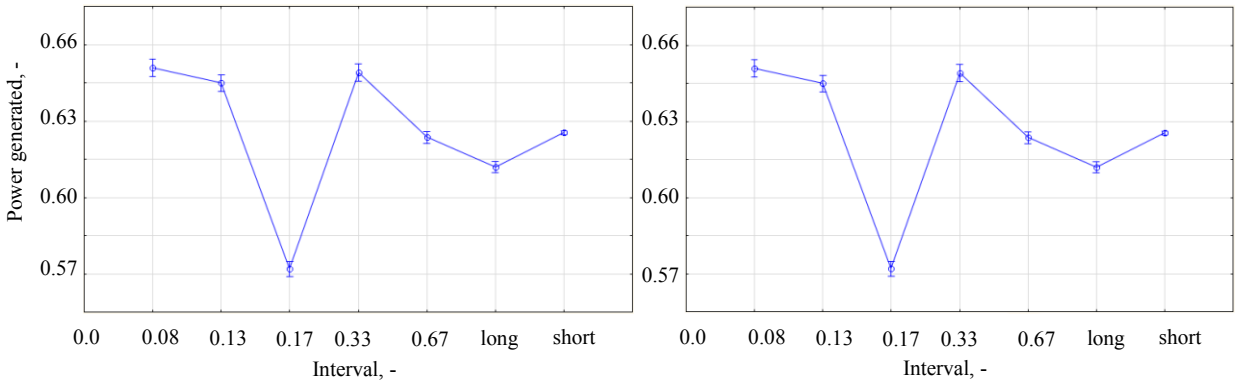

Figure 49. Influence of sootblowing interval on electrical power output: scenario I (left) and scenario II (right). [147]

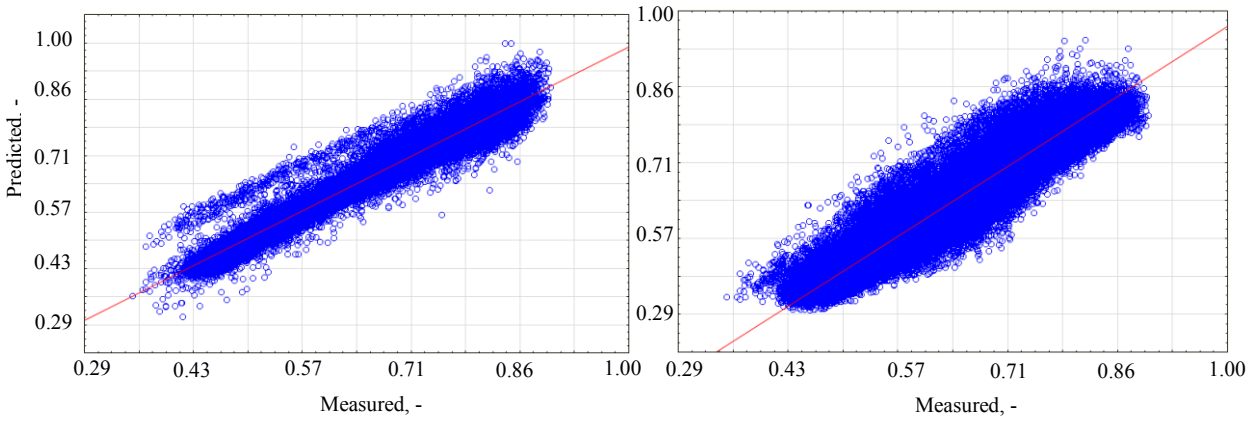

Figure 50. General linear model for predicting electrical power output: scenario I (left) and scenario II (right). [147]

Similarly to the preceding results [147], Kieliba [146] also observed that sootblowing intervals of 0.08 and 0.33 resulted into higher electrical power output than the other possibilities. Moreover, the 'short' intervals showed better power output results than the 'long' intervals (See Appendix B, Figure B-3). Although the influence of sootblowing interval on other variables was also studied [146], the results were not clear to the current author and are not discussed further in this work.

4.4.4 Correlation studies of selected variables

The correlation studies were done on two primary tasks. The first task was to investigate the influence of hot flue gas temperature, T4 and ammonia flow, q9 on NOx content in the flue gas at the stack outlet (NOx7). The other task was to examine the influence of fluidized bed temperature, T3 and the flue gas temperature, T4 on electrical power output. Two researchers studied this task and their work is summarized in this section.

Methods and testing environment

In the first work, Li [149], applied the Kasem software tools for data pre-processing as well as data analysis. Data pre-processing included visual inspection, selection of relevant data and data filtering. Other notable software tools used included Matlab 2016a and Microsoft Excel 2016. In data filtering, a median filter in Kasem software was employed. Data analysis was done using PCA method. For the first task, the variables investigated comprised of T3, T4, q8, q9, O6, V1, V2 and NOx7. In the second task, more variables were added which included T5, m5 and EP (See Table 2 for variable descriptions). Seppälä [150] also studied the same task with the help of Kasem software. Similarly to the previous works [149] data pre-processing steps included visual inspection, selection of relevant variables, data filtering and finally data correlation analysis. The list of relevant variables studied included T3, T4, q9, NOx7 and electrical power output (EP). Data filtering was implemented using two types of filters, the aberrant filter and a median filter for elimination of spikes and smoothing of data respectively. Correlation analysis was done using the Kasem software visualization tools.

Results and Discussions

The results obtained by Seppälä [150] using correlation analysis tools in Kasem, showed that flue gas temperature, T4 has no strong effect on NOx emissions (NOx7). Similarly, it was also observed that ammonia addition (q9) plays no significant role concerning NOx emissions based on the near zero correlation coefficient between the two variables. On the other hand, flue gas temperature, T4 was found to affect strongly and positively on electrical power output. However, the temperature of the fluidized bed (T3) appeared to show no influence on power output (EP).

According to Li's observations [149] from PCA analysis, T4, q9 and NOx7 were found to have a common relationship amongst each other. In fact, from the available process information [144], increase in flue gas temperature, T4 also leads to increased NOx content in the flue gas. Moreover, ammonia addition to the combustion chamber is meant for controlling the NOx content in exiting flue gas. Therefore, because NOx emissions are controlled, it partly explains the lack of a clear correlation between variables, NOx7 and T4. Concerning the influence of T3 and T4 on power output (EP), both researchers reported similar observations. Figure 51 describes the observed behavior between the studied variables.

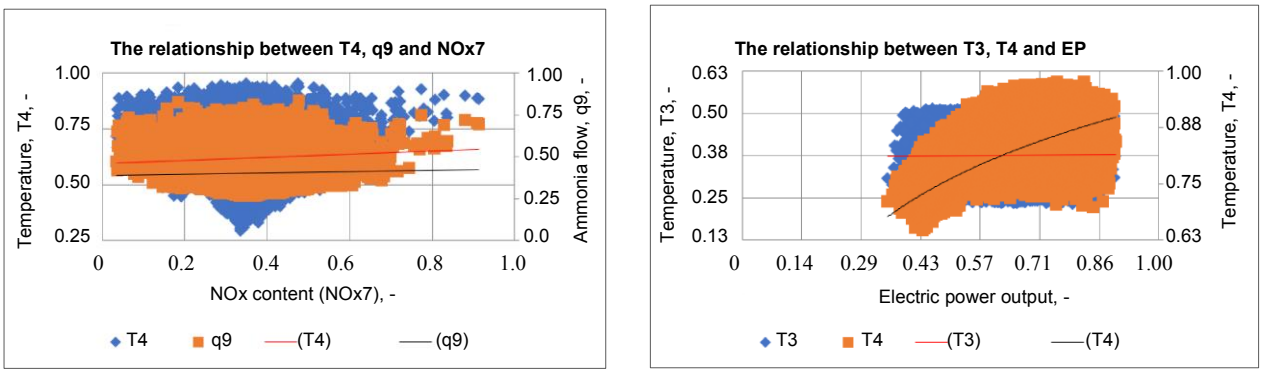

Figure 51. Visualization of the behavior of studied variables. [149]

5 Development of a process monitoring scheme

In the monitoring scheme for the waste-to-energy plant, a cloud computing environment is proposed with software tools such as Scikit-learn and H2O.ai. Notable commercially available cloud computing platforms include Microsoft Azure, Amazon web services, IBM cloud and so forth. Some of these cloud platforms for instance Microsoft Azure provide big-data processing tools as well as machine learning algorithms which can be implemented in real-time data analytics for process monitoring. One of the supported machine learning libraries in Microsoft azure is H2O.ai, which provides data pre-processing tools as well as various monitoring methods. H2O is an open source machine learning and deep learning library. In addition to machine learning algorithms, it also provides deep learning capabilities. Figure 52 illustrates a generalized process monitoring scheme which can be employed in this case study.

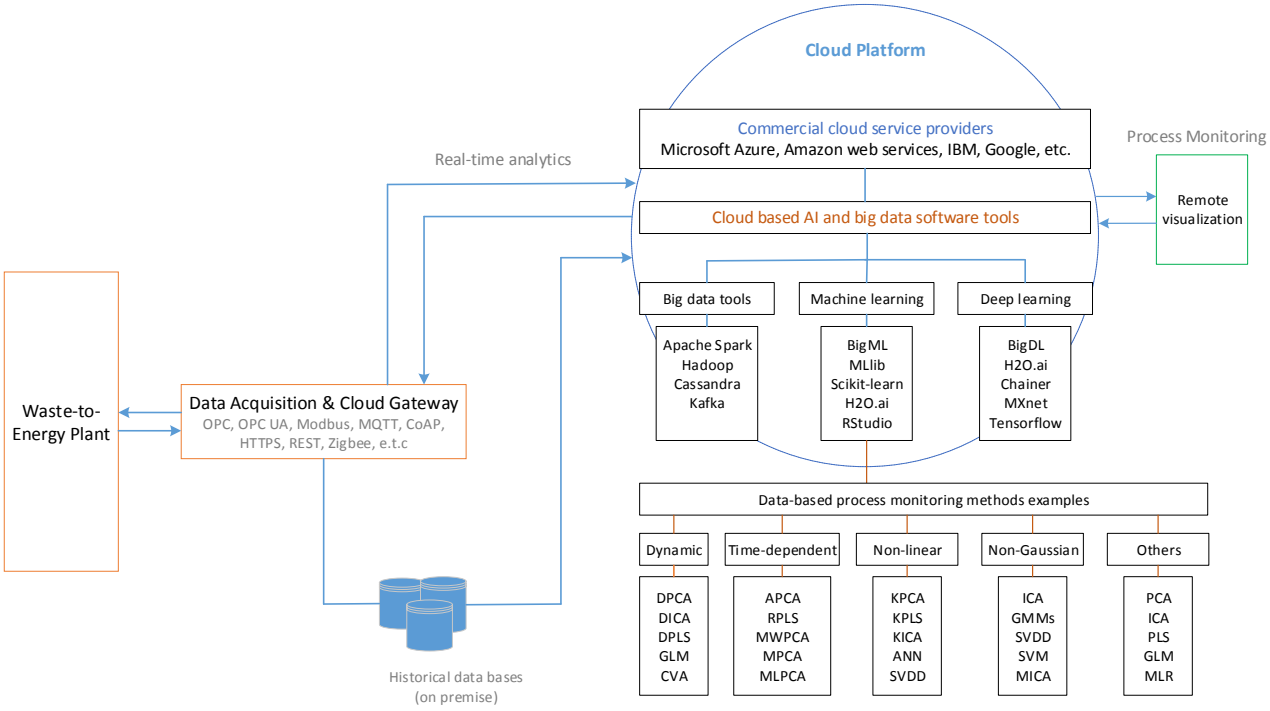

Figure 52. A general process monitoring scheme for a waste-to-energy plant

5.1 A monitoring scheme based on Microsoft Azure computing platform

As previously noted in Chapter 3.4, Microsoft Azure cloud computing platform is among the most popular PaaS currently available. Similarly to other PaaS, Microsoft Azure can be used to build, test, deploy and manage AI applications. This work proposes a real-time monitoring scheme based on Microsoft azure stream analytics with the help of azure machine learning APIs (e.g. anomaly detection, cluster models), among other azure big-data analytics services for instance Apache Storm on Azure HDInsight. In addition to conventional machine learning algorithms, Microsoft azure AI platform [151] provides access to deep learning frameworks which includes Microsoft Cognitive Toolkit,

TensorFlow, Chainer, MXNet, Keras, DIGITS, Deep Water, among others [152]. These can be utilized through Azure's 'Data Science Virtual Machines' (DSVMs). The DSVMs are pre-configured virtual machines [153] within Microsoft Azure cloud platform that are used for data analytics, machine learning and AI model development and deployment. Figure 53 illustrates the Microsoft Azure stream analytics in real-time data analytics.

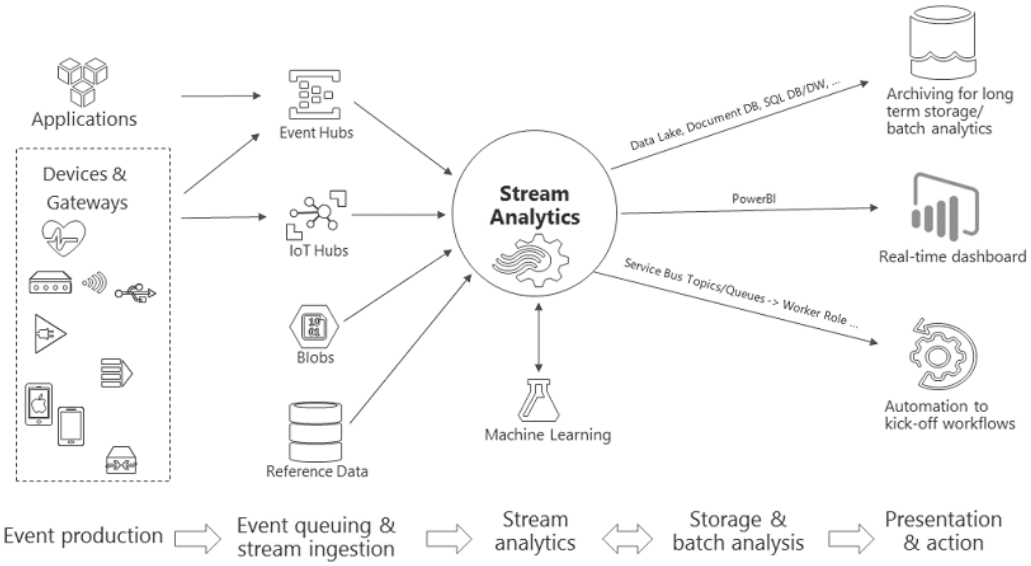

Figure 53. A simplified illustration of Microsoft Azure stream analytics. [154]

According to Figure 53, streaming data from device gateways is transmitted through either Event Hubs or IoT Hubs from where it is ingested into the Azure Stream Analytics platform. Using Azure Blob storage service also historical data can be ingested into Azure Stream Analytics. Other data sources can be connected to the analytics processing engine via the event hubs or directly such as reference (static or slow changing) data. Streamed data may be sent to data storage centers such as Azure Blob, Azure Data Lake, Azure SQL Database or Azure Cosmos DB for archiving or further perform batch data analytics using Azure HDInsight tools. Moreover, a streaming job can be visualized using Azure's PowerBI real-time dashboard or simply sent to another data analytics services via event hubs, Azure Service Bus, Queues, etc. Azure Stream Analytics uses a 'query language' to specify the input streaming data. Queries are also used to define the desired machining leaning algorithms to be applied. It is also possible to implement JavaScript user-defined functions (UDFs) within the Stream Analytics query for instance in executing complex computations.

In Figure 54, a process monitoring scheme was developed based on Microsoft Azure cloud computing platform. The scheme highlights in simple terms, data acquisition from the waste-to-energy plant, cloud gateway, Azure stream analytics engine, Azure AI platform [155], as well as real-time data visualization and other possible actions. Azure Stream Analytics is a proprietary services, scalable (based on the number of streaming units with a higher limit of 1 GB/second) and it is also integrated with Azure Machine Learning where models can be built as functions when creating a streaming job. Other than Azure Stream Analytics, open source big-data tools like Apache Storm, Apache Spark are accessible through Azure HDInsight. In addition to machine learning algorithms, more robust deep learning algorithms are available through a variety of deep learning frameworks.

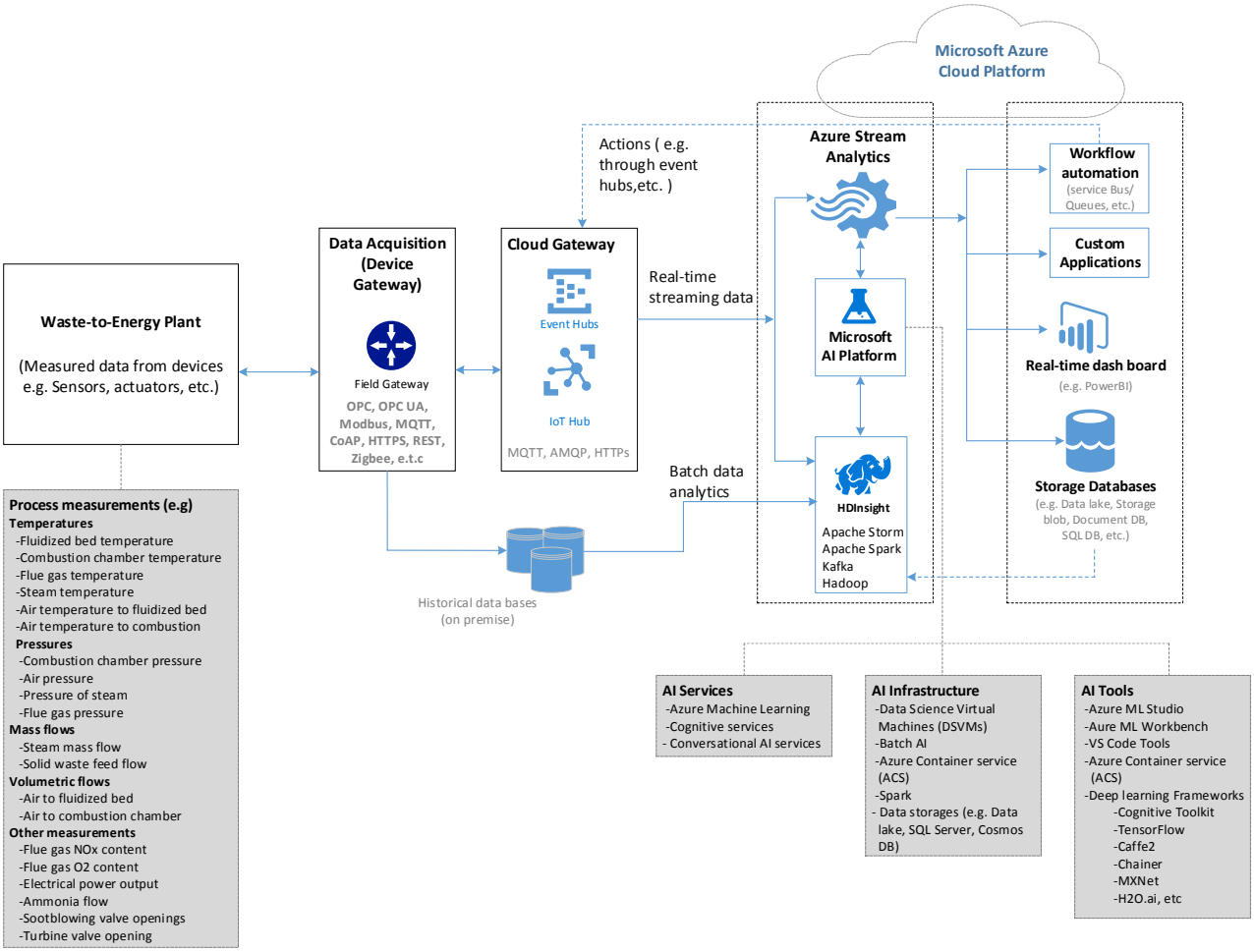

Figure 54. A proposed process monitoring scheme for a waste-to-energy plant using Microsoft Azure Cloud.

5.2 A monitoring scheme based on open source tools

In addition to commercialized cloud computing platforms and AI software tools, this work has also emphasized the availability and potential of similar but open source software tools, which can be employed in real-time big-data analytics. For instance, Cloud Foundry (PaaS) on OpenStack provides an open source alternative to proprietary PaaS clouds such as Microsoft Azure, Amazon Web services (AWS), IBM's Bluemix, Predix, among others. However, it is important to note here that open source tools may require extra effort for example during set-up, installation and network connections, before they can be ready for use. In other words, expert knowledge in many cases would be necessary to assemble and integrate different open source tools into a usable and reliable cloud computing platform. Nevertheless, in this Chapter 5.2, a process monitoring scheme based on open source tools is presented as shown in Figure 55. Like in the previous Section, the proposed scheme only highlights some of the key features of how the process is connected to the cloud computing environment.

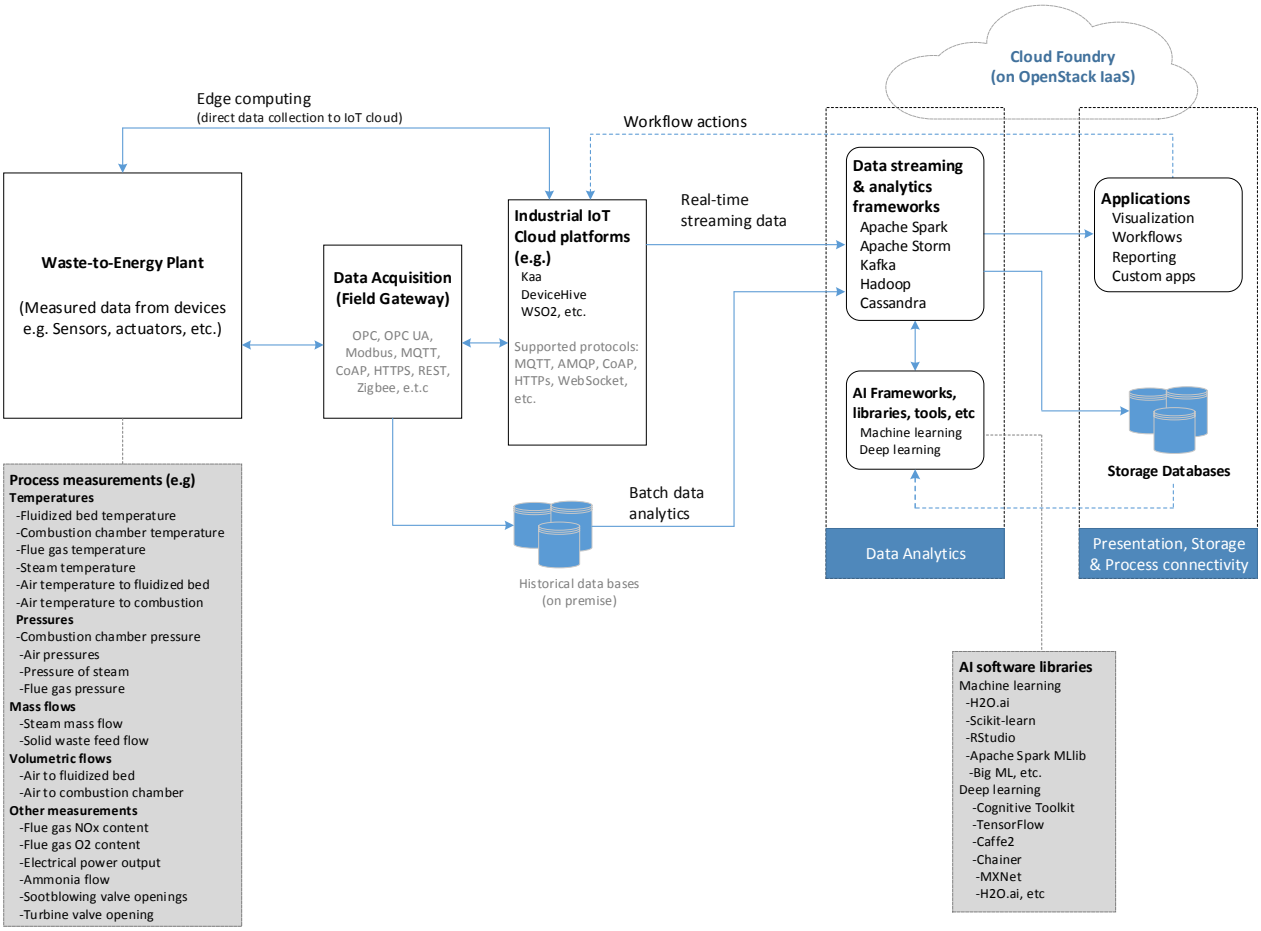

Figure 55. A proposed monitoring scheme for a waste-to-energy plant using open source AI and computing tools.

The proposed scheme (Figure 55), suggests an industrial internet of things (IIoT) cloud platform such as Kaa IoT to be applied as a link between the process data collection infrastructure and Cloud Foundry computing platform. A simplified architecture of Cloud Foundry on OpenStack can be viewed in Figure 26. Moreover, 'edge computing' may be utilized for some process devices using IoT cloud tools. The scheme also demonstrates use of real-time data streaming and analytics open source tools such as Apache Spark, Hadoop, Apache Storm, among other choices which are compatible with Cloud Foundry platform. On the other hand, there several open source AI frameworks which can be utilized in data analytics including H2O.ai, Scikit-learn, Microsoft Cognitive Toolkit, TensorFlow, Chainer and many more. Cloud Foundry also provides an environment where customer tailored applications can be deployed and implemented as desired. Similarly to Microsoft Azure, persisted data may be stored in the Cloud Foundry Blob Storage or using other external cloud databases.

As a remark, the scheme described in Figure 55, only serves as a basis for realizing an open source based real-time process monitoring system. The most suitable software components for instance in data streaming and AI tools would need conclusive experimentation which is not considered in this work.

6 Conclusions

A process monitoring scheme applicable to a waste-to-energy plant was developed based on the state-of-the-art artificial intelligence methods coupled with big-data processing tools and cloud computing technologies. The scheme emphasizes the utilization of deep learning based algorithms for analyzing process data and development of predictive analytic models useful in process monitoring. Another key aspect of the proposed monitoring scheme is the realization of real-time process monitoring in addition to offline data analysis. This is supported by the implementation of big-data processing frameworks such as Apache Spark, Hadoop, Apache Cassandra and so forth. Moreover, the work also investigated the performance of relatively simple and commonly applied data-based process monitoring methods such as principal component analysis, partial least squares, multivariable linear regression, among others in analyzing plant data. Although such methods are able to provide reliable results, their application can be limited in only certain process operational cases. For instance, according to the results for some of the studied process phenomena of the waste-to-energy plant, previous researchers observed non-linear behaviors. Such behaviors could not be described by relatively simple methods like principal component regression and partial least square regression models. Furthermore, it was demonstrated that process knowledge is important for example in the selection of most relevant process variable in order to achieve better results while employing traditional statistical methods. These noted shortcomings of the studied data-based monitoring methods can be handled by use of deep learning based models.

Deep learning algorithms present robust performance in handling non-linear, dynamic and high dimensionality systems among others. However, it is important to note that deep learning algorithms also require huge amounts of data to perform efficiently. Another drawback concerning deep learning algorithms is the high computation power requirement for training models. Nevertheless, currently there are a number of supercomputing platforms from several technology suppliers. For example, NVIDIA, Intel and IBM provide computing platforms designed for the state-of the art AI computing and big-data processing. Most of these technologies are available under cloud computing environments either as open source or proprietary cloud services. Moreover, there several cloud services which provide both machine learning and deep learning algorithms.

Although the present work was primarily tailored around monitoring a waste-to-energy industrial use case, the concept can be applied to other process industrial operations. In addition, current AI methods offer other possibilities such as learning based process control by using deep reinforcement models to determine the optimal control policies. Other areas of application of this work may include predictive maintenance of process equipment and in product quality control through development of robust soft sensors.

To summarize, this work provides a foundation for continuous development and testing of different process monitoring systems using deep learning algorithms and big data software tools. Considering the presence of various open source and commercial AI platforms, the future work is expected to select the most suitable and economical AI software tools and cloud computing platforms. The work also provides the basis of implementing latest AI tools in other areas of process operations such as process control and product quality control.

7 References

1. Turing A.M. (1950) Computing machinery and intelligence. Mind 49:433-460
2. Smith et al. (2006) The History of Artificial Intelligence. History of Computing. University of Washington. Available: https://courses.cs.washington.edu/courses/csep590/06au/projects/history-ai.pdf [September 25, 2017]
3. Gill Press (2016) A Very Short History Of Artificial Intelligence (AI). available: https://www.forbes.com/sites/gilpress/2016/12/30/a-very-short-history-of-artificial-intelligence-ai/3/#1dcf4759be7c [September 24, 2017]
4. BBC. AI: 15 key moments in the story of artificial intelligence. Available: http://www.bbc.co.uk/timelines/zq376fr [September 24,2017]
5. AAAI. AI Topics. A Brief History of AI. Available: https://aitopics.org/misc/brief-history [September 24, 2017]
6. Goodfellow I. Bengio Y and Courville A (2016) Deep learning. Cambridge. MA:MIT Press
7. Rosenblatt F (1957) The perceptron. A perceiving and recognizing automaton. Project Para. Cornell Aeronautical Laboratory.
8. Lighthill J (1972) Artificial Intelligence: A General Survey. Available: https://www.math.snu.ac.kr/~hichoi/infomath/Articles/Lighthill%20Report.pdf [September 25, 2017]
9. Yunhe Pan (2016) Heading toward Artificial Intelligence 2.0.. Artificial Intelligence-Perspective. Engineering 2. 409-413.
10. Launchbury J. A DARPA Perspective on Artificial Intelligence. Available: https://www.darpa.mil/attachments/AIFull.pdf [September 25, 2017]
11. Tzezana R. Artificial Intelligence Tech Will Arrive in Three Waves. available: https://futurism.com/artificial-intelligence-tech-will-arrive-in-three-waves/ [September 25, 2017].
12. Ross E P. (2017) Intel Buys Mobileye for $15 billion. Available: https://spectrum.ieee.org/cars-that-think/transportation/self-driving/intel-buys-mobileye-for-15-billion [September 25, 2017]
13. International Corporation data (IDC). Worldwide Spending on Cognitive and Artificial Intelligence Systems Forecast to Reach $12.5 Billion This Year. According to New IDC Spending Guide. Available: https://www.idc.com/getdoc.jsp?containerId=prUS42439617 [September 25, 2017]
14. US Government. Artificial Intelligence. Automation and the Economy. Available: https://www.whitehouse.gov/sites/whitehouse.gov/files/images/EMBARGOED%20AI%20Economy%20Report.pdf [September 25, 2017]
15. IBM (2017). ABB and IBM Partner in Industrial Artificial Intelligence Solutions. Combining ABB Ability and IBM Watson for Superior Customer Value. Available: https://www-03.ibm.com/press/uk/en/pressrelease/52301.wss [September 25, 2017].
16. Schwab K (2016) The Fourth Industrial Revolution: what it means. how to respond. Available: https://www.weforum.org/agenda/2016/01/the-fourth-industrial-revolution-what-it-means-and-how-to-respond/ [September 25, 2017].
17. What is artificial intelligence? (2016. May 18). Progressive Digital Media Technology News Retrieved from https://search.proquest.com/docview/1789725622?accountid=27468
18. Vallverdu J (2006) Choosing between different AI approaches? The scientific benefits of the confrontation. and the new collaborative era between humans and machines. tripleC 4(2): 209-216.
19. Evry. The new wave of Artificial Intelligence. Available: https://www.evry.com/globalassets/insight/bank2020/the-new-wave-of-artificial-intelligence---labs-whitepaper.pdf [September 25, 2017]
20. Blazensky E (2017) Artificial Intelligence. Machine Learning& Deep learning; A Love Triangle. Available: http://blog.voicebase.com/artificial-intelligence-machine-learning-deep-learning-love-triangle [September 25, 2017]

21. Martin (2015) Artificial Intelligence: A Complete Guide. Available: https://www.cleverism.com/artificial-intelligence-complete-guide/ [September 25, 2017]
22. Gantori S (2017) How artificial intelligence will transform Asia. Shifting Asia: Artificial Intelligence. 1-24.
23. UBS. The evolution of artificial intelligence. Available: https://www.ubs.com/microsites/artificial-intelligence/en/new-dawn.html [September 25, 2017]
24. CBINSIGNTS (2017) The 2016 AI Recap: Startups See Record High In Deals And Funding. Available: https://www.cbinsights.com/research/artificial-intelligence-startup-funding/ [September 25, 2017]
25. Economist. China may match or beat America in AI. Available: https://www.economist.com/news/business/21725018-its-deep-pool-data-may-let-it-lead-artificial-intelligence-china-may-match-or-beat-america [September 25, 2017]
26. McKinsey Global Institute (2017). Artificial Intelligence: Implications for China. Available: http://www.mckinsey.com/~/media/McKinsey/Global Themes/China/Artificial intelligence Implications for China/MGI-Artificial-intelligence-implications-for-China.ashx [September 25, 2017]
27. Jing M (2017) The future is here: China sounds a clarion call on AI funding. policies to surpass US. Available: http://www.scmp.com/tech/article/2077845/future-here-china-sounds-clarion-call-ai-funding-policies-surpass-us [September 25, 2017]
28. O'Leary E D. (2013) Artificial Intelligence and Big Data. IEEE Intelligent systems. pp.96-99
29. McKinsey Global Institute (2011) Big data: The next frontier for innovation. competition and productivity. Available: http://www.mckinsey.com/business-functions/digital-mckinsey/our-insights/big-data-the-next-frontier-for-innovation [September 25, 2017]
30. Moura A J and Serrao C. (2015) Security and Privacy Issues of Big Data. Available: https://www.researchgate.net/publication/281404634_Security_and_Privacy_Issues_of_Big_Data [September 25, 2017]
31. Russom P (2011) Big Data Analytics. TDWI Best Practices Report. Available: https://vivomente.com/wp-content/uploads/2016/04/big-data-analytics-white-paper.pdf [September 25, 2017]
32. Xue-Wen Chen and Xiaotong Lin (2014) Big data deep learning: Challenges and perspectives. IEEE Access 2:514-525
33. Zheng et al. (2017) Hybrid-augmented intelligence: collaboration and cognition. Front Inform Technol Electron Eng 18(2): 153-179
34. Predictive Analytics Today. What is Cognitive Computing? Top 10 Cognitive Computing Companies. Available: https://www.predictiveanalyticstoday.com/what-is-cognitive-computing/#content-anchor [September 25, 2017]
35. Pereira T D. Schatsky D. Sallomi P and Dalton R. (2015) Cognitive technologies in the technology sector. From science fiction vision to real-world value. Available: https://dupress.deloitte.com/dup-us-en/focus/cognitive-technologies/artificial-intelligence-in-technology-sector-tmt.html [September 25, 2017]
36. Watson Internet of Things. Available: https://www.ibm.com/internet-of-things/iot-solutions/iot-manufacturing/ [September 26, 2017]
37. Alpaydin E. (2010) Introduction to Machine Learning. Cambridge. MA:MIT Press
38. Murphy. K. P. (2012). Machine learning: A probabilistic perspective. Cambridge. MA: MIT Press
39. Louridas P and Ebert C (2016) Machine Learning. IEEE Software. pp.110-115
40. Ge, Z., Song, Z., Ding, X.S and Huang, H "Data Mining and Analytics in the Process Industry: The Role of Machine Learning," in IEEE Access, vol. 5, pp. 20590-20616, 2017. doi: 10.1109/ACCESS.2017.2756872
41. Overview diagram of Azure Machine Learning Studio capabilities. Available: https://docs.microsoft.com/en-us/azure/machine-learning/studio/studio-overview-diagram [January 5, 2018]

42. Littman L M (2015) Reinforcement learning improves behaviour from evaluative feedback. Nature. 521. pp.445-451.
43. Yuxi Li (2017) Deep reinforcement learning: An Overview. Cornell University Library. Available: https://arxiv.org/abs/1701.07274 [September 25, 2017].
44. Volodymyr Mnih et al. (2015) Human-level control through deep reinforcement learning. Nature. 518. pp.531-533
45. Silver D (2016) Deep Reinforcement Learning. Available: https://deepmind.com/blog/deep-reinforcement-learning/ [September 25, 2017]
46. LeCun Y. Bengio Y and Hinton G (2015) Deep learning. Nature. 521. pp.436-444.
47. Deng L and Yu D. (2013) Deep Learning Methods and Applications. Foundations and Trends in Signal process. 7(3-4). pp.197-387
48. Falcini, G. Lami and A. M. Costanza, "Deep Learning in Automotive Software," in IEEE Software, vol. 34, no. 3, pp. 56-63, May-Jun. 2017.doi: 10.1109/MS.2017.79
49. Deng Li (2012) Three Classes of Deep Learning Architectures and Their Applications: A Tutorial Survey. Microsoft Research. Redmond. USA.
50. Natasha Jaques et al. (2017) Multimodal Autoencoder: A Deep Learning Approach to Filling in Missing Sensor Data and Enabling Better Mood Prediction. Seventh International Conference on Affective Computing and Intelligent Interaction (ACII). IEEE.
51. Salakhutdinov R (2015) Learning Deep Generative Models. Available: https://www.cs.cmu.edu/~rsalakhu/papers/annrev.pdf [September 25, 2017]
52. Athiwaratkun B and Kang K (2015) Feature Representation In Convolutional Neural. Available: https://arxiv.org/abs/1507.02313 [September 26, 2017]
53. CS231n Convolutional Neural Networks for Visual Recognition. Available: http://cs231n.github.io/convolutional-networks/ [September 25, 2017]
54. Lee, B.K., Cheon, S and Kim, O.C. (2017) A Convolutional Neural Network for Fault Classification and Diagnosis in Semiconductor Manufacturing Processes. IEEE Transactions on Semiconductor Manufacturing, vol. 30, no. 2, pp. 135-142. doi: 10.1109/TSM.2017.2676245
55. Pascanu et al. (2013) How to Construct Deep Recurrent Neural Networks. Cornell University Library. Available: https://arxiv.org/abs/1312.6026v1 [September 25, 2017]
56. Hermans M and Schrauwen B (2013) Training and Analyzing Deep Recurrent Neural Networks. NIPS Proceedings. Available: https://papers.nips.cc/paper/5166-training-and-analysing-deep-recurrent-neural-networks [September 25, 2017]
57. Deepa. S N & Baranilingesan. I. (2017). Optimized deep learning neural network predictive controller for continuous stirred tank reactor. Computers & Electrical Engineering. .10.1016/j.compeleceng.2017.07.004.
58. Lenz I. Knepper R and Saxena A (DeepMPC: Learning Deep Latent Features for Model Predictive Control. Available: http://deepmpc.cs.cornell.edu/ [September 25, 2017]
59. Spielberg SPK. Gopaluni RB. Loewen PD. Deep reinforcement learning approaches for process control. ADCONIP. 2017:201-206. http://ieeexplore.ieee.org/document/7983780. doi: 10.1109/ADCONIP.2017.7983780.
60. Zhao et al. (2016) Deep Learning and Its Applications to Machine Health Monitoring: A Survey. IEEE Transactions on Neural Networks and Learning Systems. Available: https://arxiv.org/abs/1612.07640 [September 25, 2017]
61. Deng X. Tian X. Chen S. Harris CJ. (2017) Deep learning based nonlinear principal component analysis for industrial process fault. Neural Networks (IJCNN). 2017 International Joint Conference on. http://ieeexplore.ieee.org/document/7965994. doi: 10.1109/IJCNN.2017.7965994.
62. Yao L& Ge Z. Deep learning of semi-supervised process data with hierarchical extreme learning machine and soft sensor application. http://ieeexplore.ieee.org/document/8002611. doi: 10.1109/TIE.2017.2733448.

63. Wang J et al. (2016) Deep learning-based soft-sensing method for operation optimization of coke dry quenching process. ChiCC. 2016:9087-9092. http://ieeexplore.ieee.org/document/7554805. doi: 10.1109/ChiCC.2016.7554805.
64. Lopez R. Getting started with Neural Designer. Available: https://www.neuraldesigner.com/learning/tutorials/getting_started_with_neural_designer [September 25, 2017].
65. Beale M H. Hagan T M and Demuth B H (2017) Neural Network Toolbox™. Getting Started Guide. MATLAB.
66. Skymind. Skymind Intelligence Layer: Powering the AI Stack for Enterprise. Available: https://skymind.ai/products [September 25, 2017].
67. IBM PowerAI. Get started or get scaling faster with a software distribution for machine learning running on the Enterprise Platform for AI: IBM Power Systems. Available: https://www.ibm.com/ms-en/marketplace/deep-learning-platform [September 25, 2017].
68. Cho et al. (2017) PowerAI DDL. Available: https://arxiv.org/abs/1708.02188 [September 25, 2017]
69. VIDi. Deep Learning-based industrial image analysis. Available: https://cdn2.hubspot.net/hubfs/13219/Mfg-Catalogs/ViDi_brochure.pdf?t=1465494945428 [September 25, 2017].
70. IBM Watson. Available: https://www.ibm.com/watson/stories/woodside.html [September 25, 2017]
71. The NVIDIA DGX-1 Deep Learning System. Available: http://images.nvidia.com/content/technologies/deep-learning/pdf/Datasheet-DGX1.pdf. [September 25, 2017].
72. Intel. Deep learning Deliver Advanced Analytics for Financial Services Firms. Solution Brief. Available: https://www.intel.com/content/dam/www/public/us/en/documents/solution-briefs/deep-learning-delivers-advanced-analytics-solution-brief.pdf [September 25, 2017].
73. WOLFRAM. Neural Networks. Available: http://www.wolfram.com/language/11/neural-networks/ [September 25, 2017]
74. TensorFlow. An open-source software library for Machine Intelligence. Available: https://www.tensorflow.org/ [September 25, 2017].
75. Mo et al. (2017) Performance of Deep Learning Computation with TensorFlow Software Library in GPU-Capable Multi-Core Computing Platforms. Available: https://www.semanticscholar.org/paper/Performance-of-deep-learning-computation-with-Tens-Mo-Kim/960e381d077db63f9317bf7f37985e24ce5e40ef [September 25, 2017]
76. Theano at a Glance. Available: http://deeplearning.net/software/theano/introduction.html [September 25, 2017].
77. Keras: The Python Deep Learning library. Available: https://keras.io/ [September 25. 2017].
78. Caffe. Available: http://caffe.berkeleyvision.org/ [September 25, 2017]
79. Microsoft Cognitive Toolkit. Model Gallery. Available: https://www.microsoft.com/en-us/cognitive-toolkit/features/model-gallery/?filter=Tutorial [September 25, 2017]
80. Deep learning Software. Available: https://developer.nvidia.com/deep-learning-software [September 25, 2017].
81. Deep Learning for Java. Open-source. Distributed. Deep learning Library for the JVM. Available: https://deeplearning4j.org/ [September 25, 2017].
82. Torch. A Scientific Computing Framework for LuaJIT. Available: http://torch.ch/ [September 25, 2017]
83. Bahrampour S. Ramakrishnan N. Schott L and Shah M (2015) Comparative Study of Deep Learning Software Frameworks. Available: https://arxiv.org/abs/1511.06435 [September 25, 2017]
84. Comparing Top Deep Learning Frameworks: Deeplearning4j. Torch. Theano. TensorFlow. Caffe. Paddle. MxNet. Keras & CNTK. Available: https://deeplearning4j.org/compare-dl4j-torch7-pylearn [September 25, 2017].
85. Mxnet. Deep Learning Programming Style. Available: https://mxnet.incubator.apache.org/architecture/program_model.html [September 25, 2017].

86. Chen et al. (2015) MXNet: A Flexible and Efficient Machine Learning Library for Heterogeneous Distributed Systems. Available: https://arxiv.org/abs/1512.01274 [September 25, 2017]
87. Introduction to Chainer. Available: https://docs.chainer.org/en/stable/tutorial/basic.html#core-concept [September 25, 2017]
88. Hido S (2016) Complex neural networks made easy by Chainer. A define-by-run approach allows for flexibility and simplicity when building deep learning networks. Available: https://www.oreilly.com/learning/complex-neural-networks-made-easy-by-chainer [September 25, 2017].
89. Sergey E (2017) BigDL: Distributed Deep Learning on Apache Spark. Available: https://software.intel.com/en-us/articles/bigdl-distributed-deep-learning-on-apache-spark [September 25, 2017].
90. Wang et al. (2014) Minerva: A Scalable and Highly Efficient Training Platform for Deep Learning. Available: https://stanford.edu/~rezab/nips2014workshop/submits/minerva.pdf. [September 25, 2017].
91. Zhu H (2015) Fast and Scalable Deep Learning on the Minerva System. Master's thesis. McGill University.
92. Welcome to H2O 3. Available: http://docs.h2o.ai/h2o/latest-stable/h2o-docs/welcome.html [September 26, 2017]
93. Kejela G. Esteves M R and Rong C (2014) Predictive Analytics of Sensor Data Using Distributed Machine Learning Techniques. 2014 IEEE 6th International Conference on Cloud Computing Technology and Science. Pp.626-631.
94. Why H2O? Available: https://www.h2o.ai/h2o/ [December 13, 2017]
95. Neon. Available: https://neon.nervanasys.com/index.html/ [September 26, 2017]
96. Vadaldi A. Lenc K and Gupta A. MatConvNet: Convolutional Neural Networks for Matlab. Available: http://www.vlfeat.org/matconvnet/#documentation [September 26, 2017].
97. Lin M. Li S. Luo X and Yan S (2015) Purine: A Bi-Graph Based Deep Learning Framework. Available: https://arxiv.org/abs/1412.6249 [September 26, 2017].
98. Kai S (2016) Deep Learning Technique Predicts Gas Quality During Chemical Production Process. Available: https://www.arcweb.com/blog/deep-learning-technique-predicts-gas-quality-during-chemical-production-process [September 26, 2017]
99. IOT-Internet of Things. Artificial Intelligence: Quality Prediction of Chemical Products in the Production Process. Available: https://iot.do/artificial-intelligence-quality-prediction-chemical-products-production-process-2016-09 [September 26, 2017]
100. Cowden M (2017) 'Rebel' approach sets Big River apart: CEO. American Metal Market. Available: https://bigriversteel.com/wp-content/uploads/2017/04/Rebel-approach-sets-Big-River-apart-AMM-040717.pdf [September 26, 2017]
101. Yanchunas D (2017) Big River mill smartens up with AI technology. American Metal Market. Available: https://bigriversteel.com/resource/big-river-mill-smartens-ai-technology/ [September 26, 2017].
102. Insights Success. Noodle.ai: Pioneering Enterprise Artificial Intelligence. Available: http://www.insightssuccess.com/noodle-ai-pioneering-enterprise-artificial-intelligence/ [December 13, 2017]
103. POSCO (2017) The world's first smart factory to introduce artificial intelligence. Available: http://www.posco.com/homepage/docs/eng5/jsp/prcenter/news/s91c1010025v.jsp?idx=2707&onPage=1 [September 26, 2017]
104. CeleraOne. Surface Inspection (SI). Available: http://www.celeraone.com/en/steel-applications#Automatedqualityoptimisation [September 26, 2017].
105. Picot G (2015) How Data Science & Metallurgy meet on the IIoT. Available: http://eigeninnovations.com/eigenblog/2015/09/02/how-data-science-metallurgy-meet-on-the-iiot/ [September 26, 2017].

106. What is Cloud Computing? https://aws.amazon.com/what-is-cloud-computing/ [December 13, 2017]
107. IBM. IaaS, PaaS and SaaS – IBM Cloud service models. Available: https://www.ibm.com/cloud/learn/iaas-paas-saas [December 13, 2017]
108. What is cloud computing? Available: https://www.salesforce.com/cloudcomputing/ [December 13, 2017]
109. Cloud Computing Service Models and their Benefits. Available: https://www.msi-geek.com/7357/cloud-computing-service-models-benefits [January 3, 2018]
110. Overview of Red Hat OpenStack Platform. Available: https://access.redhat.com/documentation/en-us/red_hat_openstack_platform/12/html/product_guide/ch-rhosp-software#sect-components [January 7, 2018]
111. CloudStack Documentation: Release 4.11.0. Available: https://media.readthedocs.org/pdf/cloud-stack/latest/cloudstack.pdf [January 4, 2018]
112. Cloud Foundry Overview. Available: http://docs.cloudfoundry.org/concepts/overview.html [January 4, 2018]
113. Cloud Foundry: Application runtime. Available: https://www.cloudfoundry.org/application-runtime/ [January 4, 2018]
114. OpenShift Orign: Overview. Available: https://docs.openshift.org/latest/architecture/index.html [January 4, 2018]
115. Eclipse IoT White Paper-The Three Software Stacks Required for IoT Architectures. Available: https://iot.eclipse.org/resources/white-papers/Eclipse IoT White Paper - The Three Software Stacks Required for IoT Architectures.pdf [December 13, 2017]
116. Amazon. How the AWS IoT Platform Works. Available: https://www.amazonaws.cn/en/iot-platform/how-it-works/#device [December 13, 2017
117. Microsoft, Azure and the Internet of Things. Available: https://docs.microsoft.com/en-us/azure/iot-suite/iot-suite-what-is-azure-iot [December 13, 2017]
118. Sam George, Microsoft introduces new open-source cross-platform OPC UA support for the industrial Internet of Things. Available: https://blogs.microsoft.com/iot/2016/06/23/microsoft-introduces-new-open-source-cross-platform-opc-ua-support-for-the-industrial-internet-of-things/ [December 13, 2017]
119. Predix Developer Network: Documentation. Available: https://docs.predix.io/en-US/content/platform/get_started/predix_overview/ [December 13, 2017]
120. MindSphere: The cloud-based, open IoT operating system from Siemens. Available: https://www.siemens.com/content/dam/webassetpool/mam/tag-siemens-com/smdb/digital-factory/mindsphere/online/documents/mindsphere-iot-os-en-online.pdf [December 13, 2017]
121. Kaa IOT, Industrial Internet of Things Platform: Kaa delivers essential middleware for the Industrial Internet of Things. Available: https://www.kaaproject.org/industrial-automation/ [December 13, 2017]
122. Kaa IOT, Bring your devices to life with the Kaa IoT Platform. Available: https://www.kaaiot.io/products/platform/ [December 13, 2017]
123. DiveHive, IoT Made Easy: Open Source IoT Data Platform with the wide range of integration options. Available: https://devicehive.com/#home [December 13, 2017]
124. Mainflux. Available: http://mainflux.readthedocs.io/en/latest/ [December 13, 2017]
125. Mainflux, What is Mainflux? Available: https://www.mainflux.com/ [December 13, 2017]
126. Eclipse Kapua-Open IoT Cloud Platform. Available: https://www.eclipse.org/community/eclipse_newsletter/2017/march/article4.php [December 13, 2017]
127. Eurotech, IoT Platform Everyware Cloud: The IoT Platform for Devices, Embedded Applications and Data Management. Available: https://www.eurotech.com/en/products/iot/iot+platform [December 13, 2017]

128. Lelylan Dev Center, Architecture Microservices Explained. Available: http://dev.lelylan.com/architecture [December 13, 2017]
129. WSO2, Products Overview. Available: https://wso2.com/platform [December 13, 2017]
130. SiteWhere, System Overview. Available: http://documentation.sitewhere.io/overview.html [December 13, 2017]
131. Schiemann, R. (2017) MIDICON – Industrial use case provided by Outotec. MIDICON summer camp in Aalto University (Espoo, Finland): 29-31 August 2017
132. O'Donovan. P., Bruton, K and O'Sullivan, J.T.D (2015) An industrial big data pipeline for data-driven analytics maintenance applications in large-scale smart manufacturing facilities. Journal of Big Data (2015) 2:25
133. Introduction to data Preprocessing, lecture notes, MIDICON course 2017, Aalto University, 29-31 August 2017
134. Fuentes, G.V.J. (2017) MIDICON Assignment: Prediction of combustion chamber exit temperature for a waste to energy process. MIDICON summer camp in Aalto University (Espoo, Finland): 29-31 August 2017
135. Ge, Z., Song, Z. and Gao, F. (2013) Review of Recent Research on Data-Based Process Monitoring. Industrial & Engineering Chemistry Research 2013 52 (10), 3543-3562. DOI: 10.1021/ie302069q
136. Schach, E and Kupka, N. (2017) MIDICON Assignments 2 & 5: Industrial Use Case from an Outotec Waste to Energy plant. MIDICON summer camp in Aalto University (Espoo, Finland): 29-31 August 2017
137. Burchell, J. (2017) MIDICON Assignment: Predicting the Combustion Chamber Exit Temperature in an Advanced Stage Gasifier. MIDICON summer camp in Aalto University (Espoo, Finland): 29-31 August 2017
138. Silva, M.A.C. (2017) MIDICON Assignment: Predict combustion chamber exit temperature. MIDICON summer camp in Aalto University (Espoo, Finland): 29-31 August 2017
139. Nuorivaara, T. (2017) MIDICON Assignment: A study on the influences of the flue gas temperature in an Outotec waste to energy gasification plant. MIDICON summer camp in Aalto University (Espoo, Finland): 29-31 August 2017
140. Ai, M. (2017) MIDICON Assignment: Predictive model for combustion chamber exit temperature. MIDICON summer camp in Aalto University (Espoo, Finland): 29-31 August 2017
141. Wang, P and Wang, H. (2017) MIDICON Assignment: Prediction model for flue gas temperature. MIDICON summer camp in Aalto University (Espoo, Finland): 29-31 August 2017
142. Penkov, S.A. (2017) MIDICON Assignment: Prediction model for flue gas temperature. MIDICON summer camp in Aalto University (Espoo, Finland): 29-31 August 2017
143. Patscheider, S and Weingrill, G. (2017) MIDICON Assignment: Development of a soft sensor for syngas heating value of a gasification plant. MIDICON summer camp in Aalto University (Espoo, Finland): 29-31 August 2017
144. Schiemann, R. (2017) MIDICON – Solution propositions for assignments provided by Outotec. MIDICON summer camp in Aalto University (Espoo, Finland): 29-31 August 2017
145. Catalina, C.J. (2017) MIDICON Assignment: Study on the selection of an optimized sootblowing interval for a staged gasification power system. MIDICON summer camp in Aalto University (Espoo, Finland): 29-31 August 2017
146. Kieliba, I (2017) MIDICON Assignment: Optimize sootblowing interval. MIDICON summer camp in Aalto University (Espoo, Finland): 29-31 August 2017
147. Kossakowska, K. (2017) MIDICON Assignment Optimalisation of sootblowing interval. MIDICON summer camp in Aalto University (Espoo, Finland): 29-31 August 2017
148. Kubik, R. (2017) MIDICON Assignment: Optimize sootblowing interval. MIDICON summer camp in Aalto University (Espoo, Finland): 29-31 August 2017

149. Li, S. (2017) MIDICON Assignment: Correlation study. MIDICON summer camp in Aalto University (Espoo, Finland): 29-31 August 2017
150. Seppälä, P. (2017) MIDICON Assignment: Correlation study. MIDICON summer camp in Aalto University (Espoo, Finland): 29-31 August 2017
151. Microsoft AI platform: Artificial intelligence productivity for every developer and every scenario. Available: https://azure.microsoft.com/en-us/overview/ai-platform/?v=17.42 [January 13, 2018]
152. Deep Learning and AI frameworks. Available: https://docs.microsoft.com/en-us/azure/machine-learning/data-science-virtual-machine/dsvm-deep-learning-ai-frameworks [January 13, 2018]
153. Pre-Configured environments in the cloud for Data Science and AI Development. Available: https://azure.microsoft.com/en-us/services/virtual-machines/data-science-virtual-machines/ [January 13, 2018]
154. Introduction to Stream Analytics-Microsoft Docs. Available: https://docs.microsoft.com/en-us/azure/opbuildpdf/stream-analytics/TOC.pdf?branch=live [January 15, 2018]
155. Microsoft AI Platform: Build Intelligent Software. Available: https://azure.microsoft.com/media-handler/files/resourcefiles/5467086f-7c5a-4106-8615-7f5e4ad217fd/microsoft-ai-platform-whitepaper.pdf [January 15, 2018]

Appendix A: Tables

Table A-1. Coefficients of models

Parameter	Eqn.(2), (Fuentes, 2017)	Eqn.(3), (Nuorivaara, 2017)	Eqn.(4), (Nuorivaara, 2017)
a_0	2511.3095	122.1274	972.4451
a_1	0.3064	4.5555	4.5380
a_2	-0.8407	-0.0668	-0.0727
a_3	-2.2033	1.0666	-1.1859
a_4	-83.6384	-0.0007	-0.0096
a_5	-173.2470	-0.0010	-0.0017
a_6	8.1276	-0.6683	-0.1024
a_7	276.7384	10.9327	0.1035
a_8	9.8540	-0.0707	0.0158

Table A-2. Box-cox linear model transformation results (Schach& Kupka, 2017).

Box-cox transformation parameters	
Variable	λ
q2	0.06742434
q8	0.05367379
q9	-1.62635391
P2	-4.48521576
T4	-2.4767855
Coefficients of the linear model	
Variable parameter	Values
Intercept	4.037367e-01
q2	3.133852e-12
q8	2.772270e-10
q9	2.019505e-05
P2	NA (no result)

Table A-3. Effect of sootblowing interval on other selected variables (Catalina, 2017)

Sootblowing interval	**Observed change in average values of variables after sootblowing, %**			
	T5	**T6**	**ΔT= (T4-T6)**	**ΔT=(P4-P6)**
0.08	0.39	-1.71	1.47	2.66
0.13	0.41	-3.48	1.99	1.10
0.17	0.68	-2.74	1.91	2.55
0.33	2.08	-8.22	5.07	2.73
0.67	2.76	-15.06	11.32	-10.80

Appendix B: Figures

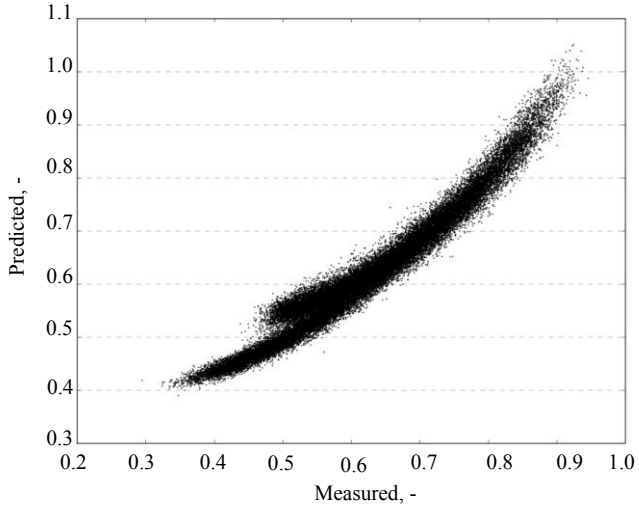

Figure B-1. Results for a PLS regression model in predicting flue gas temperature, T4 (Burchell, 2017).

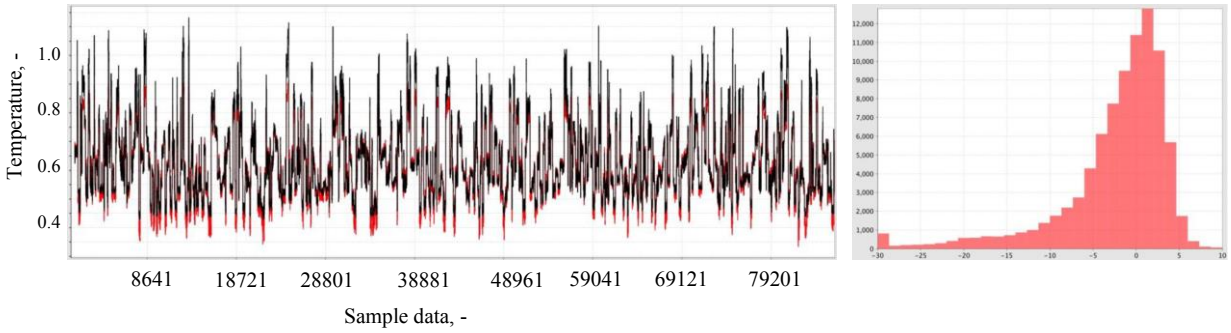

Figure B-2. Performance of a multiple iterative regression model: predictive model and residual error (Ai, 2017).

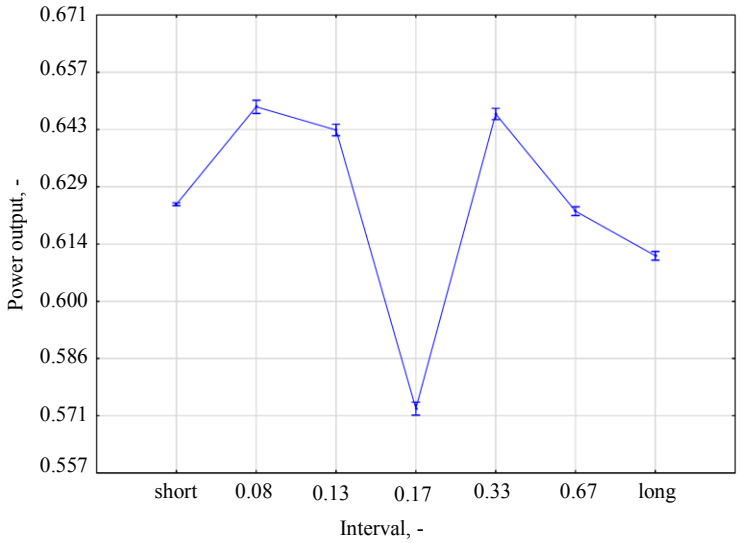

Figure B-3. Relationship between sootblowing interval and electrical power output (Kieliba, 2017).