Shubhabrata Datta
J. Paulo Davim   *Editors*

Machine Learning in Industry

Springer

Management and Industrial Engineering

Series Editor

J. Paulo Davim, Department of Mechanical Engineering, University of Aveiro, Aveiro, Portugal

This series fosters information exchange and discussion on management and industrial engineering and related aspects, namely global management, organizational development and change, strategic management, lean production, performance management, production management, quality engineering, maintenance management, productivity improvement, materials management, human resource management, workforce behavior, innovation and change, technological and organizational flexibility, self-directed work teams, knowledge management, organizational learning, learning organizations, entrepreneurship, sustainable management, etc. The series provides discussion and the exchange of information on principles, strategies, models, techniques, methodologies and applications of management and industrial engineering in the field of the different types of organizational activities. It aims to communicate the latest developments and thinking in what concerns the latest research activity relating to new organizational challenges and changes world-wide. Contributions to this book series are welcome on all subjects related with management and industrial engineering. To submit a proposal or request further information, please contact Professor J. Paulo Davim, Book Series Editor, pdavim@ua.pt

More information about this series at http://www.springer.com/series/11690

Shubhabrata Datta · J. Paulo Davim
Editors

Machine Learning in Industry

Springer

Editors
Shubhabrata Datta 🔘
Department of Mechanical Engineering
SRM Institute of Science and Technology
Chennai, Tamil Nadu, India

J. Paulo Davim 🔘
Department of Mechanical Engineering
University of Aveiro
Aveiro, Portugal

Preface

Machine learning (ML) is a method for training computers or making the computer learn automatically from the supplied information or data. Different methods of machine learning originate from the nature and follow the principle of biological learning. The applications of ML in the real world are increasing fast and encompass our daily life without our knowledge. The manufacturing and other industrial sectors have also started using the ML in their plants effectively. With the advent of the concept of Industry 4.0, the pace of application of ML in core industries will only increase.

The introductory chapter of this book describes the basic concepts of the most popular techniques of machine learning. It explains the different classes of machine learning approaches and describes the statistical and artificial intelligence-based machine learning techniques in brief. The techniques like decision tree, linear regression, least square method, artificial neural network, clustering techniques, and deep learning are discussed.

Practical examples of applications and case studies using practical industrial problems or problems relevant to the industries are described in the rest of the chapters, i.e., Chapters Neural Network Model Identification Studies to Predict Residual Stress of a Steel Plate Based on a Non-destructive Barkhausen Noise Measurement–Performance Improvement in Hot Rolling Process with Novel Neural Architectural Search. In the Chapter Neural Network Model Identification Studies to Predict Residual Stress of a Steel Plate Based on a Non-destructive Barkhausen Noise Measurement, a ML-based procedure using artificial neural network and genetic algorithm is proposed for predicting the residual stress in steel. Chapter Data Driven Optimization of Blast Furnace Iron Making Process Using Evolutionary Deep Learning deals with modeling the complicated process of iron and steel making. Parameters like burden distribution, oxygen enrichment, productivity improvement, composition of the top gas, and many other parameters in blast furnace control the productivity in a complex way. In this chapter, machine learning approaches are employed to model the blast furnace process and evolutionary algorithms are employed to optimize the process.

In the Chapter A Brief Appraisal of Machine Learning in Industrial Sensing Probes, the author describes the method to implement ML in the digital control and

monitoring systems in industries. How the sensors can be used effectively for the integration of industrial data for standard ML is discussed. The application of ML for searching the root cause of sliver defects in the cold rolling mill is described in the Chapter Mining the Genesis of Sliver Defects Through Rough and Fuzzy Set Theories. A typical case is described where the genesis of the defect is analyzed using two methods, viz. rough set and fuzzy set theories. This chapter shows how the rough set theory can be used to select the important variables to which the cause of the defect can be attributed. The rules created from the data are used in the fuzzy framework for developing a predictive model. An overview of the application of ML methods in the field of materials science with respect to materials—processes—knowledge formalization is provided in the Chapter Machine Learning Studies in Materials Science.

ML can be used for developing surrogate models or metamodels, which can replace complex analytical models and numerical simulations for optimization, sensitivity analysis and uncertainty quantification. In Chapter Accurate, Real-Time Replication of Governing Equations of Physical Systems with Transpose CNNs — for Industry 4.0 and Digital Twins, Convolutional-NN-like architectures are used as surrogate models on two different applications of reduced Navier–Stokes equations containing high nonlinearities and abrupt discontinuities.

Deep learning is the latest paradigm of machine learning. The Chapter Deep Learning in Vision-Based Automated Inspection: Current State and Future Prospects of the book evaluates the application of the techniques for vision-based automated inspection. Here a detailed discussion on the merits and demerits of deep learning for automated inspection tasks in industries are made. The ninth and last chapter proposes a novel algorithm to design multi-layered feed forward neural networks with parsimony as well as accuracy using multi-objective optimization.

It is quite evident that the authors of the present book covered various aspects and applications of machine learning relevant to the industry. The editors express their sincere gratitude to the authors for their excellent contributions. The editors also express their thanks to all the reviewers who have contributed immensely to the improvement of the quality of the chapters. Both the editors are grateful to their colleagues, friends, and family members. The editors also acknowledge the Springer team for their excellent work towards shaping the compilation beautifully.

Chennai, India   Shubhabrata Datta
Aveiro, Portugal   J. Paulo Davim

Contents

About the Editors

Shubhabrata Datta presently a Research Professor in the Department of Mechanical Engineering, SRM Institute of Science and Technology, Chennai, India, did his Bachelors, Masters, and Ph.D. in Engineering from Indian Institute of Engineering Science and Technology, Shibpur, India (previously known as B.E. College Shibpur) in the field of Metallurgical and Materials Engineering. Dr. Datta has more than 28 years of teaching and research experience. His research interest is in the domain of design of materials using artificial intelligence and machine learning techniques. He was bestowed with the Exchange Scientist Award from the Royal Academy of Engineering, UK and worked in the University of Sheffield, UK. He also worked in the Dept of Materials Science and Engineering, Helsinki University of Technology, Finland, Dept of Materials Science and Engineering, Iowa State University, Ames, USA, and Heat Engineering Lab, Dept of Chemical Engineering, Åbo Akademi University, Finland as Visiting Scientist. He is a Fellow of Institution of Engineers (India), Associate Editor, Journal of the Institution of Engineers (India): Series D, and editorial board member of several international journals.

J. Paulo Davim received his Ph.D. degree in Mechanical Engineering in 1997, M.Sc. degree in Mechanical Engineering (materials and manufacturing processes) in 1991, Mechanical Engineering degree (5 years) in 1986, from the University of Porto (FEUP), the Aggregate title (Full Habilitation) from the University of Coimbra in 2005 and the D.Sc. from London Metropolitan University in 2013. He is Senior Chartered Engineer by the Portuguese Institution of Engineers with an MBA and Specialist title in Engineering and Industrial Management. He is also Eur Ing by FEANI-Brussels and Fellow (FIET) by IET-London. Currently, he is a Professor at the Department of Mechanical Engineering of the University of Aveiro, Portugal. He has more than 30 years of teaching and research experience in Manufacturing, Materials, Mechanical, and Industrial Engineering, with special emphasis in Machining & Tribology. He also has interest in Management, Engineering Education, and Higher Education for Sustainability. He has guided large numbers of postdoc, Ph.D., and master's students as well as has coordinated and participated in several financed research projects. He has received several scientific awards. He has worked as evaluator of projects for ERC-European Research Council and other international research

agencies as well as examiner of Ph.D. thesis for many universities in different countries. He is the Editor in Chief of several international journals, Guest Editor of journals, book Editor, book Series Editor, and Scientific Advisory for many international journals and conferences.

Fundamentals of Machine Learning

A. Vinoth and Shubhabrata Datta

Abstract This introductory chapter describes the techniques of machine learning. Primarily the concept of machine learning in the context of artificial intelligence and data analytics is explained. The application process of the above to big data is introduced. Classification of machine learning approaches is described. Then some of the variedly used statistical and artificial intelligence-based machine learning techniques are described in brief. The techniques discussed include decision tree, linear regression, least square method, artificial neural network, clustering techniques. The concepts of deep learning are also introduced.

Keywords Machine learning · Artificial intelligence · Data analytics · Supervised learning · And unsupervised learning

1 Introduction

Machine learning (ML) is a subdivision of computational science which is progressed from the learning of data classification based on the gained understanding and also from the learning gained on computational-based principles of Artificial Intelligence (AI). In simple, machine learning is training the computers to learn automatically through the inputs deprived of being explicitly programmed [1]. The term learning evolved from the humans and animals. Animal and machine learning have quite a few matches. Indeed, a lot of methods in machine learning originate to mark principles of animal and human learning by computational models. For instance, habituation is a basic scholarly conduct where an animal step by step quits reacting to a rehashed stimulus. Dogs are considered to be a perfect example for animal learning where it is capable of substantial learning if it is trained to perform various activities like rolling over, sitting and picking up the things, etc.

A. Vinoth · S. Datta (✉)
Department of Mechanical Engineering, SRM Institute of Science and Technology, Kattankulathur, Chennai 603203, Tamil Nadu, India
e-mail: shubhabp@srmist.edu.in

With regard to the former example of effective learning, there are few examples that could demonstrate machine learning where we use in our day-to-day life of modern era. Virtual personal assistants, traffic predictions using GPS navigation, surveillance of multiple cameras by AI to detect the crime or unusual behaviour of people, social media uses ML for face recognition and news feed personalization, search engine result refinement, e-mail spam filtering where a machine memorize all the earlier labeled spam e-mails by the user, and lot more applications where ML is widely in use. Through all these applications, it is understood that the incorporation of prior knowledge will preference the mechanism of learning. ML is also closely interconnected to computational statistics where it familiarizes the prediction making [2]. Anyone might wonder 'why a machine must learn something?' There are few aims why ML is essential. Obviously we have just referenced that the accomplishment of learning in machines may assist us with seeing how creatures and people learn. Yet there are few essential engineering details that persist and some of these are

- Certain tasks cannot be clearly explained without example; i.e. we may have the option to identify input/output sets however not a brief correlation between inputs and preferred outputs.
- It is probable that there are unseen relationships between inputs and outputs among huge loads of data. Machine learning methods can repeatedly be utilized to reveal these relationships.

When do we want machine learning instead of straight away program our computers to perform a task? Two characteristics of a certain problem may demand for the usage of programs that learn and develop on the basis of their experience/understanding, i.e. the complexity of problem and the want for adaptivity. There are tasks that are complex to program, for instance human activities like driving, understanding of images and voice recognition of a person, etc., where the art of ML works on the principle of learning through experience that could yield reasonable results [3]. One restraining feature of automated tools is their inflexibility, i.e. once the coding has been formulated and installed, it remains unchanged. Still, many tasks change over period or from one end user to another. For such problems, the utilization of ML which has coding that decode the earlier written program adapting a fixed program to check the variations among the styles of different users.

2 Artificial Intelligence

Artificial intelligence (AI) denotes the replication of human intellect in machines that are encoded to imitate human activities. The term may likewise be applied to any machine that exhibits human qualities, for instance, learning and critical thinking [4]. A more elaborate definition describes AI as 'a system's capacity to effectively decipher outside information, to gain from such information, and to utilize those learning to accomplish explicit objectives and assignments through adaptable transformation'. As innovation progresses, earlier standards that branded AI become

outdated. For example, machines that establish necessary capabilities or identify text through model character identification are not, at this juncture said to represent AI, because this purpose is presently underrated as a built-in function of a computer. Artificial intelligence is ceaselessly evolving for the benefit of various enterprises. Machines are wired using a cross-disciplinary approach that includes arithmetic, software engineering, semantics, brain science and a lot more with specialized fields like artificial study of the mind. The objectives of AI incorporate learning, thinking, communication and recognition.

AI is exceptionally focused, and is intensely divided into subfields that are very different from one another [5]. A part of the classification is because of social and cultural elements: subfields have developed over specific foundations and contributions of various researchers. AI is additionally isolated by limited specific topics. Some subfields emphasize on the solution of explicit issues. Others center around one of a few potential procedures or on the utilization of a specific tool or toward the accomplishment of particular applications. AI has been the subject of good faith however has withstood alluring difficulties. Now, it has become a basic aspect of the innovation business, giving the truly difficult work to a significant amount of the major testing disputes in software work.

In the early nineteenth century, AI research is evolved in different ways like digital computer's formal thinking that could mimic any possible demonstration of numerical derivation in 1943, writing simple programs/algorithms to solve problems in algebra, theorems and speaking English in 1956 [6]. US government has started investing on AI research in 1960 on developing various laboratories around the world. Due to the lot of failure that has occurred in the research till 1974, finance for AI projects was difficult to obtain. During 1980s, with the help of few professionals AI research was rejuvenated by the profitable achievement of expert systems. In 1990s and the early twenty-first century, AI attained its great feats where it is utilized for logistics, data mining, clinical findings and numerous different regions all through the innovation business. The quest for more proficient critical thinking algorithms for solving problems in sequential steps is a high need for AI research. AI has made some progress on mimicking these sorts of processes which highlight on the need of good reasoning skills, neural net exploration endeavors to recreate the structures inside the cerebrum that offer ascent to this ability; measurable ways to deal with AI copy the probabilistic nature of the human capacity to predict. AI frequently spins around the utilization of algorithms where plenty of clear-cut directions that a computer can perform. An unpredictable algorithm is regularly based on other easier algorithms which basically cover deduction, reasoning and problem-solving.

Key researches of AI are knowledge depiction and knowledge engineering [7]. Huge numbers of the subjects machines are trusted upon to illuminate will need extensive information about the world. Effort on the development of AI research works is based on the commonsensical knowledge involve huge extents of lengthy ontological engineering as they should be worked, by hand, each convoluted idea in turn. The common traits of an AI system involve the following viz planning, learning, communication, perception, motion and manipulation. Pertaining to planning, an intelligent agent can imagine the future to make predictions in a better way that would

change the world and will be able to utilize the available choices to the maximum. Periodic checks on the predictions with the actuals to be done and if required an agent can make change on the plan to avoid any uncertainty. Learning involves the machine learning under three different regimes as supervised learning, unsupervised learning and reinforcement learning and the same will be discussed in detail in the latter part. Communicating the machines will be done through natural language processing where a machine can peruse and comprehend the languages that people talk. A typical technique for processing and pull out significance from normal language is done over semantic ordering which increases the processing speed and cut short the cost of large data storage. Perception of machine is the capacity to use response from various sensors to presume features of the world. Motion and Manipulation in AI are closely related to the field of robotics to handle different jobs as object management and triangulation through robots.

Long-term goals relating to AI research are (a) Societal Intelligence (b) Creativeness and (c) Common Intelligence [8]. Affective computing is the form of societal intelligence that focuses on the investigation and improvement of systems and gadgets that can perceive, decipher, measure and recreate human effects. A branch AI tends to imagination both theoretically (from a philosophical and mental viewpoint) and essentially (by way of explicit usage of charters that yield results that can be regarded as inventive, or frameworks that recognize and survey creativity). Associated zones of computational assessment are Artificial instinct and Artificial reasoning. Numerous analysts believe that their effort will eventually be merged into a machine with general intelligence (known as solid AI), all the aptitudes above and exceeding human abilities all things considered or each one of them. A couple accepts that human highlights comparable to fake alertness or an artificial brain might be necessary for such an undertaking.

Different approaches of AI are classified broadly as '1. Cybernetics and mind simulation' which connects the nerve system, theory of information and automations. '2. Symbolic AI' that would ultimately prosper in building a machine with artificial general intelligence that evolve from 1960s to 1990s as cognitive simulation (based on cognitive and management science), logic-based approach (based on the principle of abstract reasoning and problem solving), knowledge-based approach (based on knowledge revolution into AI applications), sub-symbolic approaches (to definite AI problems), computational intelligence and soft computing (subset of AI that focuses on neural networks, fuzzy systems, evolutionary computation, etc.) and statistical approaches (based on refined mathematical tools to resolve precise sub problems). Integration of the above said approaches are also possible by utilizing the 'Intelligent agent paradigm' and 'Agent and cognitive architectures' [9]. Former one focuses on considering specific glitches and discover resolutions that are valuable, without concurring on one single methodology. Latter focuses on connecting the multiple AI systems as a hybrid system, which has both symbolic and sub-symbolic components.

Different tools of AI involves search algorithm (informed and uninformed search algorithms), mathematical optimization (simulated annealing, random optimization, blind hill climbing and beam search), evolutionary algorithms (ant colony and particle
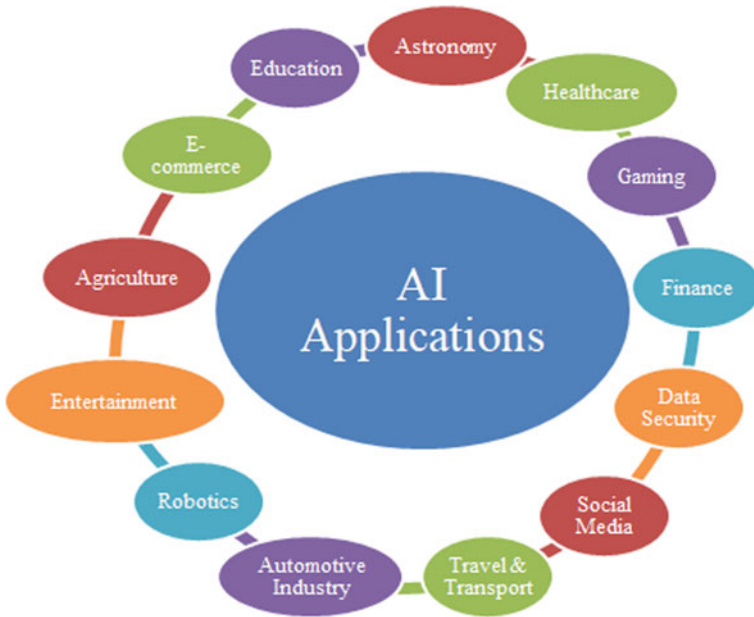
Fig. 1 Applications of Artificial Intelligence

swarm optimization, genetic algorithms and genetic programming), logic programming and automated reasoning (default logics, non-monotonic logics and circumscription), probabilistic methods for uncertain reasoning (Bayesian inference algorithm, decision networks, probabilistic algorithms, etc.), classifiers and statistical learning methods (Neural networks, Gaussian mixture model, decision tree, etc.). Figure 1 describes the applications of AI in various fields.

3   Data Analytics

One of the mathematical and statistical approaches of analyzing data is data analytics that mainly focuses on what the data can say us outside the proper modeling or testing of hypothesis. Data Analysis practices business intelligence and analytics models. Business intelligence (BI) is a prearrangement of techniques and tools for the change of crude data into significant and useful data for business research drives. BI advancements are fortified for dealing with formless data to distinguish, generate and in any case create fresh key business openings. The aim of BI is to consider the guileless understanding of huge bulks of data. One of the analytic models of data analysis is exploratory data analysis (EDA) for data investigation, and to arrive at a hypotheses that could prompt new data assortment and investigations. EDA is remarkable in relation to initial data analysis (IDA), which hubs around scrutinizing

Table 1 Techniques in EDA

Graphical techniques in EDA	Quantitative techniques in EDA
• Pareto chart • Histogram • Run chart • Stem-and-leaf plot • Box plot • Targeted projection pursuit • Multi-vari chart • Parallel coordinates • Scatter plot • Multilinear PCA • Multidimensional scaling • Odds ratio • Principal component analysi	• Ordination • Trimean • Median polish

suppositions essential for model fitting and theory testing, considering missing qualities and changing factors varying. In 1961, Tukey characterized data analysis as procedures for evaluating data, policies for cracking the outputs of such methods, means of positioning the data to simplify analysis exactly, and all the hardware and results of (numerical) statistical data put on to evaluate the data [6]. These statistical developments, all braced by Tukey, were envisioned to add-on to the scientific theory of testing measurable assumptions, chiefly the Laplacian convention's prominence on exponential families [10]. The objectives of EDA are to propose hypothesis, evaluate the expectations statistically, choosing suitable statistical tools/techniques and to pave a way for further data gathering via studies or experimentations. Some of the graphical and quantitative techniques in EDA are listed as given in Table 1.

3.1 Types of Data Analytics

Data analytics is a wide arena of study. The four essential classes of data analytics as descriptive, diagnostic, predictive and prescriptive analytics. Each of it has an alternate objective and a better position in the course of the evaluation of data. These are also the key data analytics in commercial applications. *Descriptive analytics* supports answering studies concerning the whereabouts. These methods total up enormous datasets to portray outcomes to associates. By creating key performance indicators (KPIs), these methodologies can support path achievements or dissatisfactions. Measurements, for example, return on investment (ROI) are utilized in numerous ventures. Exacting dimensions are fashioned to trail accomplishment in unambiguous ventures. This cycle needs the collection of substantial data, organizing of the data, investigation of data and conception of data. This cycle gives basic knowledge into past accomplishments.
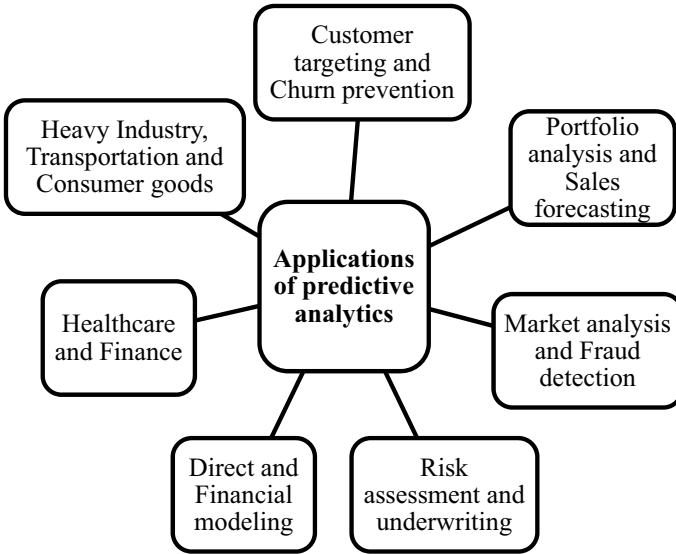
Diagnostic analytics supports answering inquiries regarding why things occurred. These methods complement more fundamental descriptive analytics. They contemplate the discoveries from descriptive analytics and burrow further to discover the reason. The performance pointers are additionally explored to find the reason for improvement. This happens mostly in three stages:

- Recognize irregularities in the data. The sudden variations in a quantity or a definite market.
- Gathering of data connected to such irregularities.
- Statistical methods are employed to ascertain networks and designs that simplify such irregularities.

Predictive analytics supports answering inquiries concerning the later events. Such methods employ verifiable data to distinguish drifts and resolve if they are possibly going to recur. Predictive analytical tools provide vital understanding into future events and it measures integrate a collection of numerical and AI procedures, for example, neural networks, decision trees and regression. The prediction of data is crucial in data analytics hence a detailed study on this is highly needed. The types of predictive analytics are 1. Predictive modelling 2. Descriptive modeling and 3. Decision modelling. Predictive models will be replicas of the link between the explicit output of a unit in a model and known credits or highlights of the unit. The goal of the model is to appraise the prospect that a comparative unit in a substitute model will display the specific output. This model is used in wide area, such as marketing, carrying out calculations in live businesses to guide a decision, crime scene investigation [11] and due to its computing quickness it can simulate behaviour or reactions of people for particular situations. Descriptive models measure influences in data to arrange clients or forecasts into collections. Not all predictive models emphasize on forestalling a self-contained client conduct, (for example, credit hazard), descriptive models distinguish an array of connections amongst clients or items. It categorizes the clients by their preferences of items and life phase rather than by the probability of clients on considering a specific task as in predictive models. Decision models depict the connection amongst all the components of a decision, the identified information (accounting outputs of predictive models), the decision, and the predicted outcomes of the decision so as to anticipate the outputs of decisions including numerous factors. Such models may be utilized in optimization, boosting definite results while limiting others. Figure 2 depicts the applications of predictive analytics in various fields broadly as businesses, science and industrial applications. Also, the methods and procedures employed to perform predictive analytics can generally be clustered into regression techniques and machine learning techniques. Further classifications of regression and machine learning practices are listed as follows in Table 2. A few of the listed techniques will be discussed further in detail.

Numerous predictive analytics tools are available both as an open-source tools (KNIME, Open NN, Orange and GNU Octave, etc.) and commercial tools (MATLAB, Minitab, STATA, SAP, Oracle data mining, etc.) [12] that are useful in processes decision making and integrating it into different operations.

Fig. 2 Applications of predictive analytics

	Regression techniques	ML techniques
Table 2 Classifications of regression and machine learning	• Linear regression model • Discrete choice model • Logistic regression • Multinomial logistic regression • Logit vs probit • Time series models • Probit regression • Classification and regression trees • Survival or duration analysis • Multivariate addaptive regression splines	• Radial basis functions • Multilayer perceptron (MLP) • Other Neural Networks • K-nearest neighbours • Naive bayes • Geospatial predictive modeling • Support vector machines

Prescriptive analytics supports answering inquiries regarding what has to be completed. By making use of bits of information from predictive analytics, that ended with data-driven selections. This licenses organizations to stick to choices in the face concerning susceptibility. Prescriptive analytics procedures rely on AI techniques that can ascertain proposals in huge datasets. By exploring previous selections and cases, the prospects of various results can be evaluated.

3.2 Data Mining

One of the activities of data analysis is 'Data mining'. Data mining (the investigation phase of the 'Knowledge Discovery in Databases—KDD' process), a multidisciplinary subspecialty of software engineering nothing but the computational method of determining patterns in massive data sets comprise approaches at the connection of artificial intelligence, machine learning, statistics and database systems [13]. One can confuse data analysis with data mining. The main difference between these two are as follows:

- Data mining recognizes and finds a shrouded design in enormous datasets whereas data Analysis gives bits of knowledge or testing of hypothesis or model from a dataset.
- Data mining is one of the events in data analysis. Data analysis is a comprehensive set of events that deals with the assortment, planning and displaying of data for mining expressive understandings or information. Both are at times comprised as a subdivision of Business Intelligence.
- Data mining educations are typically on organized data. Data analysis should be possible on organized, semi-organized or unorganized data.
- The objective of data mining is to create data more practical while data analysis aids in demonstrating a theory or taking business choices.
- Data mining need not bother with any biased theory to distinguish the example or pattern in the information. Then again, data analysis tests a given theory.
- Data mining depends on numerical and logical methods to recognize patterns or drifts wherein data analysis utilizes business intelligence and analytics models.

Data mining includes six collective modules of tasks as Anomaly detection (finding unfamiliar data sets), Association rule learning (search of correlation between variables), Clustering (act of determining sets and assemblies in data), Classification (act of simplifying known structure to new data), Regression (identifying a task that prototypes the data with the minimum blunder) and Summarization (giving an added solid depiction of the data set, comprising conception and documentation). The wide range applications of data mining focus on human rights, games, science and engineering, medical data mining, sensor data mining, visual data mining, spatial data mining, surveillance, music data mining, pattern mining, knowledge grid, temporal data mining, business and subject-based data mining.

4 Big Data

Big data is an area which breaks down ways, deliberately isolate data from, or in any case achieve data sets that are markedly massive or multifaceted to be accomplished by conventional data-processing application software. Big data encounters apprehending data, storage, data investigation, search, sharing, moving, representation,

querying, refreshing, data protection and source. The word big data repeatedly states just to the practice of predictive analytics or further definite innovative approaches to excerpt worth from data, and seldom to a precise size of data set. Precision in big data may possibly pave a way to more assured decision-making and enhanced conclusions can mean better operative efficiency, cost declines and reduced threat. Researchers, business chiefs, clinical experts, publicizing and governments routinely meet troubles with enormous data sets in regions encompassing Internet look, fintech, metropolitan informatics, and business informatics. Scholars face restraints in e-Science work, which includes meteorology, genomics, connectomics, intricate material science models, science and ecological study. The world's innovative per-capita ability to store data has largely grown like clockwork since the 1980s as of 2012; consistently 2.5 Exabytes (2.5×10^{18}) of information [14] were made as seen from Fig. 3. The trial for enormous undertakings is guessing out who should claim big data activities that ride the entire connotation.

Big data is a group of data from several sources, frequently described by the 3Vs namely Volume (quantity of data), Variety (data category) and Velocity (the rate at which the data generation happens). Over time, other Vs namely Veracity (quality of captured data), Value (business worth of the collected data) and Variability (inconsistency that hinders the process) have been added to descriptions of big data. Data management is quite a complex process when huge amount of data reaches from various sources. To offer valuable understanding to the data management and
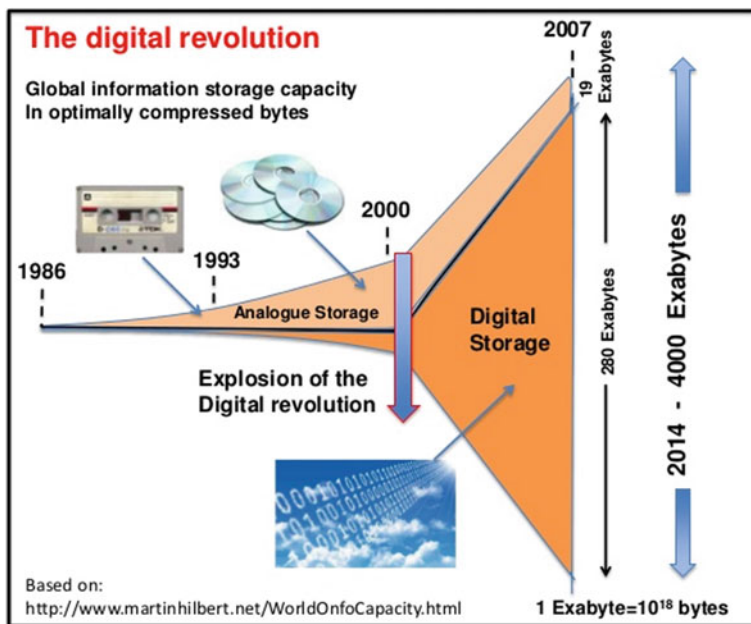
Fig. 3 Evolution and digitization of global data storage capacity (Reproduced from M. Hilbert and P. López, 2011) [14]
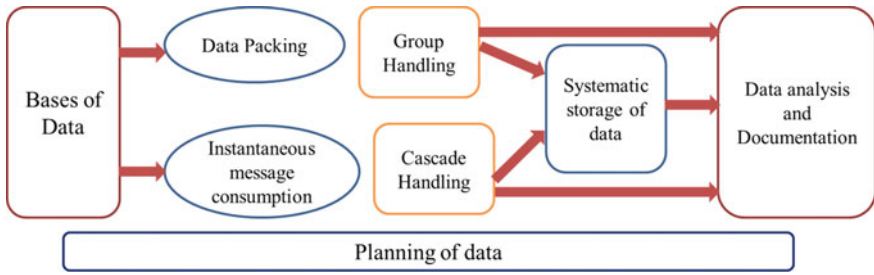
Fig. 4 Architecture of Big Data

increase correct content, data has to be handled with modern tools (analytics and algorithms) to produce expressive information.

Big data architecture denotes the rational and physical arrangement that commands how an extraordinary amount of data are consumed, processed, stored, accomplished and retrieved. Big data architecture is the base for big data analytics. The architecture mechanisms of big data analytics naturally contain four rational layers and execute four key processes as depicted in Fig. 4. 1. Big Data Sources Layer (managing both batch and real-time processing of big data such as data warehouses, SaaS applications and Internet of Things (IoT) devices), 2. Management & Storage Layer (receiving, converting and storing of data to the suitable format of data analytics tool), 3. Analysis Layer (extraction of business intelligence or BI from the storage layer) and 4. Consumption Layer (collects outputs from the analysis layer and presents them to the appropriate BI layer) [15].

Big data has originated numerous applications in several areas. The key areas where big data is being utilized are as follows. Government and private sectors, Social media analytics, Technology, Fraud detection, Call center Analytics, Banking, Agriculture, Marketing, Smartphones, Education, Manufacturing, Telecom and healthcare.

5 Supervised Learning

Supervised learning involves concluding a function from labeled training data using machine learning activity. The training data contains a set of training samples. In supervised learning, every sample is a duo involving an input item (usually a vector) and a preferred output value (also known as supervisory signal). A supervised learning algorithm examines the training data and creates a contingent function that may be utilized for representing new samples. An ideal situation allows the algorithm to properly define the class labels for hidden occurrences. This needs the learning algorithm to simplify from the training data to hidden occurrences in a 'realistic' way. In order to resolve an assigned difficulty of supervised learning, the following steps are to be followed.

1. Identifying the type of training samples
2. Collection of training set
3. Identifying the input feature illustration of learned function
4. Identifying the structure of learned function and the suitable learning algorithm
5. Completion of design on running the algorithm with the collected training set
6. Assessing the correctness of the learned function.

Four key concerns to be considered in supervised learning are (i) *Bias-variance tradeoff* [10]—A learning algorithm with small predisposition should be 'flexible' so as to perfectly fit the data. But if the learning algorithm is excessively supple, it will suit every training data set in a different way, and hence have high variance, (ii) *Function complication and volume of training data*—this issue concerns about the quantity of available training data with the complication of the function (classifier or regression), i.e. simpler the function needs a learning from small quantity of data wherein complex function requires massive quantity of training data, (iii) *dimensionality of the input space*—it depends on the dimension of the input feature vectors since extra dimensions can complicate the learning algorithm that will have more variance and (iv) *Noise in the output values*—this issue concerns about the amount of noise in the preferred output values. If the output values are improper owing to man-made or sensor errors then matching the training samples will not be efficient leads to overfitting. There are numerous algorithms in use to determine the noise in the training samples preceding to the supervised learning algorithm.

In general, all machine learning algorithms have a common principle in which it works that is they are defined as learning a target function (f) that maps the input (X) with the output values (Y) and making it predict Y for a new value of X and the relation is given as follows in Eq. (1).

$$Y = f(X) + e \tag{1}$$

There will also be an error (e) which is independent of X and this error is considered to be an irreducible error no matter how good we get the target function. A supervised learning algorithm also works on this principle. The most extensively used learning algorithms are linear regression, naive Bayes, logistic regression, Support Vector Machines, k-nearest neighbor algorithm, Neural Networks (MLP), decision trees, linear discriminant analysis and Similarity learning.

The various applications of supervised learning are widely in use in major areas such as Bioinformatics, Cheminformatics, Database marketing, Handwriting recognition, Information extraction, Pattern recognition, Speech recognition, Spam detection, Downward causation in biological system and object recognition in computer vision, etc.

6   Unsupervised Learning

Unsupervised learning is a sort of AI that searches for formerly hidden configurations in a data set with no prior labels and with at least manual oversight. Contrary to supervised learning that typically utilizes human-labeled data, unsupervised learning otherwise called as self-association takes into consideration displaying of likelihood densities over data sources [16]. In unsupervised learning, the two fundamental techniques that are utilized namely cluster analysis and principal component analysis. Cluster analysis is utilized in unsupervised learning to gather, or partitioned datasets with common features owing to generalize algorithmic connections. Cluster analysis is a subdivision of machine learning that assembles the data that has not been named, ordered or classified. Rather than reacting to feedback, cluster analysis recognizes unities in the data and responds depending on the existence or nonexistence of such unities in each fresh portion of data. This methodology aids identify abnormal data points that do not apt for any group.

A fundamental use of unsupervised learning is in the area of density estimation in statistics; however unsupervised learning incorporates a lot of areas relating to briefing and clarifying data features. In contrast to supervised learning that uses conditional probability distribution $p_x (x \mid y)$ trained on the y of input data whereas unsupervised learning uses a priori probability distribution $p_x (x)$.

Number of most general algorithms are used in unsupervised learning and each approach practices numerous techniques as follows: 1. Clustering (ex: OPTICS algorithm, k-means, hierarchical clustering, etc.), 2. Irregularity detection (ex: local outlier factor and isolation forest), 3. Neural networks (ex: autoencoders, hebbian learning, deep belief nets, etc.) and 4. Approaches for learning latent variable models (ex: method of moments, expectation-maximization or EM algorithm, blind signal separation techniques, etc.) [6].

7   Reinforcement Learning

Reinforcement learning (RL) is a part of machine learning that dealt with how software agents should take movements in a setting to exploit the idea of accumulative return. It is one amongst the three common machine learning models, along with supervised learning and unsupervised learning. In comparison with supervised learning, RL doesn't require labeled input or output values and also not in need of sub-optimal activities to be adjusted rather it aids in identifying stability between the investigation of unexplored area and manipulation of existing knowledge. Due to its simplification, reinforcement learning is considered in various disciplines like game theory, control theory, operations research, information theory, simulation-based optimization, multi-agent systems, swarm intelligence and statistics. For instance, in operation research and control literature, RL is termed as neuro-dynamic programming. The glitches of attention in RL had been deliberate in the optimal control theory

where it mainly focuses on the presence and categorization of optimal solutions and precise computation of algorithms.

Reinforcement learning is mainly compatible to glitches that contain a long-term versus short-term prize trade-off [17]. It is proven to be effective for many problems, comprising robot control, elevator scheduling, telecommunications, backgammon and checkers. Two factors that create reinforcement learning influential, i.e. the usage of samples to enhance performance and the usage of function approximation to deal with huge settings. A simple reinforcement learning model comprises of:

1. A group of environment states;
2. A group of actions;
3. Guidelines for movement between states;
4. Guidelines that define the scalar instant return of a movement; and
5. Guidelines that designate the observation of agent.

8 Decision Tree

A decision tree is a supporting technique that helps to make decisions using a kind of tree-like prototype of choices and their possible significance, comprising chance event results, source costs, and service. It showcases a supervised machine learning algorithm that has only restricted statements. Decision trees are generally utilized in operations research, especially in decision analysis, to support finding an approach most probable to attain an objective, but are also a widely used tool in machine learning and data mining [18]. This technique aims to make a model which forecasts the worth of a target variable/output depending on various input parameters. Das et al. reported hot rolled steel plate classification [19], obtained from CART analysis, for quality check based on chemical composition. A similar decision tree showing composition-based classification of strength of mild steel is given in Fig. 5.

Any tree model will have a *root node* that helps to divide the data into two or more sets. The key attribute of this node is chosen by using attribute selection measure (ASM) technique. *Branch* is the portion of the complete decision tree otherwise named as sub-tree. Arrowheads are used to distribute a node to two or more sub-nodes depending on if-else conditions and the process is called *splitting*. *Decision*

Fig. 5 Optimal decision tree for mild steel plates for classifying low (1) and high strength
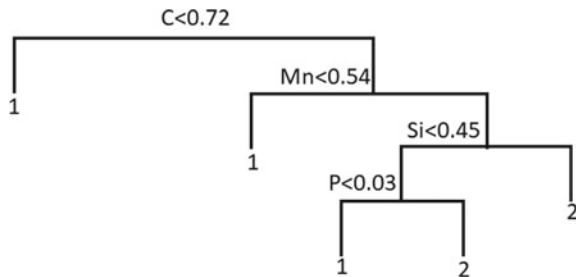
node is the one on splitting the sub-nodes into successive sub-nodes. *Leaf or terminal node* is the conclusion of the decision tree in which a sub-node can't be split further. *Pruning* is the process of eliminating a sub-node from a tree.

Decision trees that are utilized in data mining are of two kinds. Tree models in which the target variable can yield a fixed set of values known as *classification trees.* Using these tree models, leaves of the tree signify class labels and branches of the tree signify combinations of sorts that initiate those class labels. Tree models where the target variable can utilize continuous values (usually real numbers) are called *regression trees.* In the analysis of decision, a decision tree can be utilized to visually and clearly denote decisions and decision making. In case of data mining, a decision tree defines data but not decisions; instead for decision making, the subsequent classification tree shall be made use of as an input. The combination of the above two kinds under a single roof is termed as Classification and Regression Tree (CART).

In order to reduce the data used in data mining, ASM technique is widely in practice that helps various algorithms for finding the best attributes. Two key types of ASM techniques are *Gini index and information gain.* Gini Index is the quantity of degree of possibility of a specific variable that is categorized incorrectly. The mathematical relation of a Gini index is given in Eq. (2)

$$Gini = 1 - \sum_{i=1}^{n}(p_i^2) \tag{2}$$

where p_i denotes the probability of classifying an object in a specific class. Normally a feature with a minimum Gini index is chosen if Gini index is used as the condition for an algorithm. Information gain or ID3 algorithm is the one that helps to reduce the level of entropy from root node to leaf node that aid to identify an attribute which yields thorough evidence about a class and the same has been expressed in Eq. (3).

$$E(s) = \sum_{i=1}^{c}(-p_i log_2 p_i) \tag{3}$$

where p_i denotes the probability of entropy 'E(s)'. Normally a feature with a maximum ID3 gain is utilized as the root for splitting. Some of the notable decision tree algorithms under a wide classification are Conditional Inference Trees, ID3 (Iterative Dichotomiser 3), MARS, C4.5 (successor of ID3), CHAID (CHi-squared Automatic Interaction Detector), CART (Classification and Regression Tree), etc.

9   Least Squares

The least squares method is a statistical technique to identify the best fit for a set of data points by reducing the entirety of the squares of the residuals of points from the curve. It is a typical method in regression analysis that predicts the performance of dependent variables in relation to the independent variables. The utmost significant application is in data fitting. The best fit in the least squares limits the entirety of squared residuals being the difference between an observed value and the fitted value provided by a model. At the point when the issue has generous uncertainties in the independent variable (X variable), at that point simple regression and least squares techniques have issues; in such cases, the approach needed for fitting errors-in-factors models are deemed better than that for least squares.

In regression analysis, dependent variables are plotted on the y-axis, while independent variables are plotted on the x-axis. These descriptions will give the equation for the line of best fit as depicted in Fig. 6, which is determined from the least squares method. In contradiction of a linear problem that has a definite solution, a non-linear least squares problem has no definite solution and is usually resolved by iteration on approximating it as a linear one. Polynomial least squares define the difference in a predicted value of the dependent variable as an independent variable function and the deviances from the fitted graph. The least square methods developed in the areas of
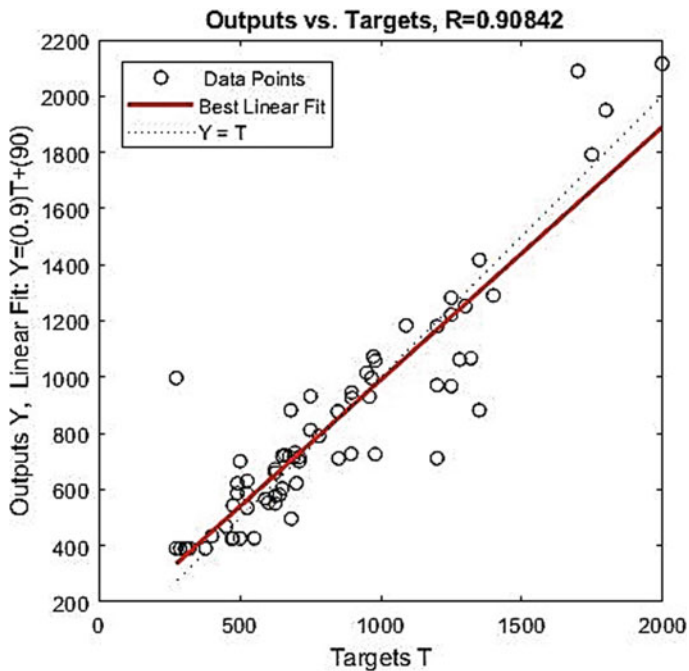

Fig. 6 Best linear fit

astronomy and geodesy through the course of the eighteenth century, where experts and statisticians wanted to deliver answers to the experiments of circumnavigating the Earth's oceans in the Era of Investigation. In 1795, German mathematician Carl Friedrich Gauss revealed the process of the least squares method but only in 1805, it was first printed by a French mathematician Adrien-Marie Legendre who described it as a numerical method for fitting linear equations to data on demonstrating a new procedure on evaluating the similar data as Laplace for the world's shape. However after 1809, Gauss brought in a new development on the method of least squares with the principles of probability, probability density, normal distribution and method of estimation. In 1810, with the base of Gauss's work, Laplace come up with Central limit theorem and in 1822, Gauss formulated Gauss-Markov Theorem [6]. Likewise many researchers have come up with various ways of implementing least squares. A problem will be defined based on an objective function which has 'm' adjusting variables of a model function defined by vector 'β' to best fit a 'n'data set that contains 'x_i' independent variable and 'y_i' dependent variable. The fit of the model is given by residuals 'r_i' as follows which is the difference between the actual values of 'y' to the predicted value of 'y'as given in Eq. (4).

$$r_i = y_i - f(x_i, \beta) \tag{4}$$

The least square methods determine the optimum variable by bringing down the sum 'S' of squared residuals and is shown below in Eq. (5)

$$S = 1 - \sum_{i=1}^{n} (r_i^2) \tag{5}$$

The formulation of the regression has its limitations as to considering only the observational errors in the dependent variable. There are two relatively dissimilar situations with different inferences as Regression for prediction and Regression for fitting an 'accurate relation'. The common ways for cracking the least square problem are by linear least squares and non-linear least squares [20]. The difference between linear least squares and non-linear least squares are 1. The model function, f, in LLSQ (linear least squares) is a linear arrangement of variables of the form $f = x_{i1}\beta_1 + x_{i2}\beta_2 + \ldots$ The prototype may signify a straight line, a parabola or any other linear combination of functions. In NLLSQ (nonlinear least squares) the variables seem as functions, such as β^2, $e^{\beta x}$ and so forth. If the derivatives $\frac{\partial f}{\partial \beta_j}$ are one as constant or the other be influenced by only on the independent variable, the variables shows the model is linear. Else it is a nonlinear model. 2. Want primary values for the variables to identify the result to a NLLSQ problem; LLSQ doesn't necessitate them. 3. In LLSQ the result is distinctive, but in NLLSQ there may be numerous minima in the sum of squares. A distinct set of global least squares termed *weighted least squares* happens when the whole of the off-diagonal entries of the residual's correlation matrix become void; the differences of the observations may even be uneven.

10 Linear Regression

Linear regression is a method for exhibiting the correlation between a dependent variable (y) and one or many independent variables (x) [21]. A model that has only one independent variable is known as simple linear regression and in case of more than one independent variable it is known as multiple linear regression. It is merely different from multivariate linear regression which expects various associated dependent variables instead of single dependent variable. Linear regression emphasizes on the restricted probability distribution of the independent variables given by the function of the model instead of the combined probability distribution of all those variables nothing but the area of multivariate analysis. It has several everyday applications involving both statistics and machine learning due to its models based on the linearly unknown variables that can fit easily rather than the models with non-linear variables and also it is at ease to find the numerical properties of the subsequent estimators. There are several techniques that train the linear models and the most familiar is known as least squares but there are other approaches of fitting the model, for example, Ordinary least squares or Gradient descent or L^1 regularization and L^2 regularization [6]. Hence least squares and linear model are diligently related but not identical in meaning.

A linear regression model is represented by a linear equation that connects a particular set of input variables (x) that gives the results of predicted output (y) for the set of x. A coefficient 'β' is allotted to each of its inputs in a linear equation as a scale factor. Also, an added supplementary coefficient called bias coefficient or intercept that provides the line as in Fig. 6 which is an example for a simple regression line has a degree to move freely on a 2D plot. A typical regression equation with one input and an output is given in Eq. (6)

$$y = \beta_0 + \beta_1 x \tag{6}$$

The complication of a linear regression model depends on the number of coefficients used in it. For example, if a coefficient is zero then it neglects the effect of that input variable and thereafter the prediction of the model. This is common in regularization methods which could modify the algorithm to simplify the complexity of the model on making an entire size of the coefficients to zero.

One has to take a call before interpreting the outcomes of the regression model where each of them may follow the *unique effect* (estimated change in the predicted output for a change in a single input where all the other covariates are held stable) or the *marginal effect* (entire derivative of predicted output relating to input). There are two possibilities that may occur while interpreting the results of regression, i.e. where if the marginal effect is huge then the unique effect is zero or if the unique effect is huge then the marginal is zero. However, there is a chance of failure for multiple regression analysis to have the correlation between the predicted output with the input since the unique effect deals with a multifaceted system where lot of interconnected constituents influences the input variable.

There are lot of additions of linear regression have been established which involves simple and multiple linear regression, general linear models, generalized linear models (GLMs), heteroscedastic models, hierarchical linear models, measurement error models, and so on. It is essential to estimate the parameter and implication in linear regression. Few of the broad estimation approaches are Least squares estimation (ex: ordinary least squares, Generalized least squares, percentage least squares, total least squares, etc.), Maximum likelihood estimation (Ex: ridge regression, lasso regression, adaptive estimation, least absolute deviation) and other miscellaneous estimation approaches (ex: Bayesian linear regression, principal component regression, Quantile regression and Least angle regression). Linear regression has its major applications in the field of finance, economics, environmental science and epidemiology in order to define the suitable correlations between the parameters.

11 Neural Networks

A chain of algorithms that replicate the actions of a human brain to describe the correlation between numerous set of data is called Neural Network (NN). Architecture of neural network is same as that of human brain's which has 'Neurons' may be a biological or artificial neurons acting as a numerical function that gathers and categorizes data in relation to a particular architecture [22]. Since 1943 till late 2000, neural networks have shown tremendous development in artificial intelligence. The evolution of NN follows as right from a computational model called threshold logic on the basis of algorithms and mathematics that focuses on brain's genetic processes and application of NN to AI. Later a Hebbian learning based on hypothesis was created and applying it with B-type machines that follow the unsupervised learning [11]. After which use of calculators as computational machines that mimic the Hebbian network was created. A two layer computer learning network algorithm was created for the recognition of pattern followed by the development of back propagation algorithm in machine learning which sorted out the issue of NN in solving the processing of circuit with mathematical notation and processing power of earlier computers. Other evolution like support vector machines and few easier methods like linear classifiers surpassed NN in machine learning admiration. Later deep learning has transformed a new attention in neural networks. Since 2006 and till date, further the developments of NN is incredible in the new era of digital computing such as feedforward NN, long short-term memory (LSTM) in pattern recognition, traffic sign recognition, molecules identification for new drugs and so on.

In order to solve artificial intelligence (AI) problems, a neural network with artificial neurons called artificial neural network (ANN) is used [23]. Each network has a solid similarity to statistical methods like curve fitting and regression analysis. Layers (input, hidden and output) of interrelated nodes constitute a basic artificial neural network as shown in Fig. 7. Alike multiple linear regression, every node of a network called a perceptron that converts as a nonlinear activation/transfer function by passing the signal given by a multiple linear regression, i.e. a neuron of ANN
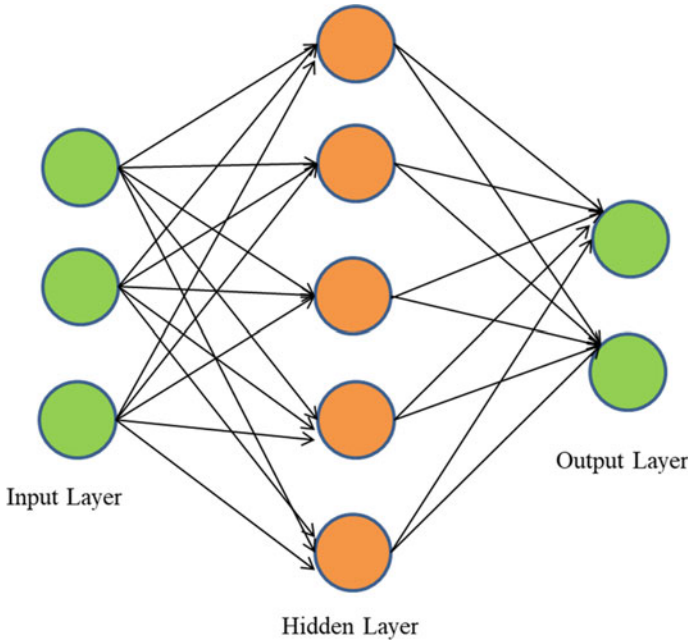
Fig. 7 Architecture of ANN

takes a signal probably a real number then processes the same and gives signal to the neurons attached to it.

In the course of learning process, neurons and the connectors generally have a weight on each input, i.e. product of inputs and weights in a transfer function that increases or decreases the signal's strength with a threshold level and the signal passes only if the cumulative signal crosses the threshold. There are different transfer functions in use and few of them are hard limit transfer function, pure linear transfer function, log-sigmoid and Tan-sigmoid transfer functions, etc. Numerically, f(x) is a function of a neuron which is a structure of another function g(x) and that can further be a structure of other functions which is completely denoted as a structure of a network that shows the relations between the variables. A typically used structure of a function is the weighted sum non-linear function which is given by the relation as in Eq. (7),

$$f(x) = U\left(\sum_{i=0}^{n} w_i g_i(x)\right) \tag{7}$$

where U is the activation function such as tan-hyperbolic. Learning of ANN is under three key paradigms namely supervised learning, unsupervised learning and reinforcement learning which was explained in detail in the earlier sessions. The training of neural networks are done by utilizing the widely used methods like simulated

annealing, particle swarm optimization, expectation maximization, evolutionary methods, genetic programming and non-parametric methods. The applications of ANN are broadly into the following categories: data processing (filtering, clustering, etc.), robotics (guiding prosthesis and manipulators), classification (recognition of sequence and pattern), regression analysis/function approximation (fitness modeling and approximation, prediction of time series), control (process control, computer numeric control and vehicle control) and computational and notional neuroscience.

12 Cluster Analysis

Cluster analysis is an approach that is utilized to arrange set of data/articles into related collections/groups named clusters. It is otherwise known to be classification analysis or clustering or numerical taxonomy. Here, there is no earlier data available about the group or cluster relationship for any of the articles. In clustering, objects are isolated into groups (clusters) with the aim that each object is more like the different objects within the same cluster rather than the objects outside the cluster [6]. Figure 8 depicts the clustering of iris flowers based on petal length and width grouped into different colors. Cluster analysis includes the problem formulation, choosing a separation measure, choosing a clustering method, finalizing the amount of clusters, interpretation of clusters and lastly evaluating the strength of clustering. Therefore cluster analysis can be expressed as a problem of multi-objective optimization that involves an iterative process of detecting knowledge with lot of trials and letdowns rather than the automatic process.

The concept of a 'cluster' may not be exactly distinct, which calls in for various clustering algorithms which vary meaningfully in their understanding/properties of
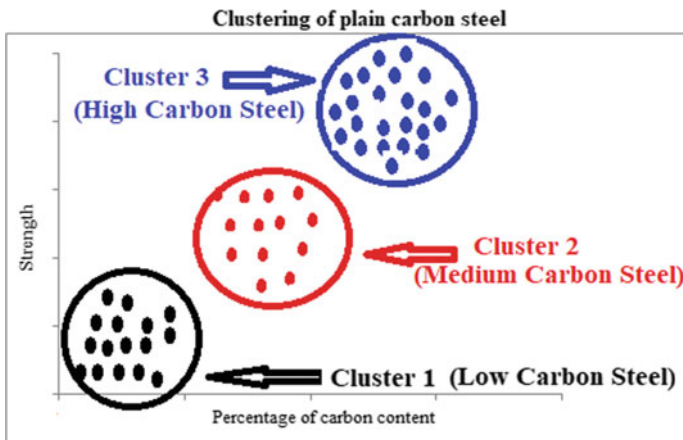
Fig. 8 Typical Cluster analysis of plain carbon steel
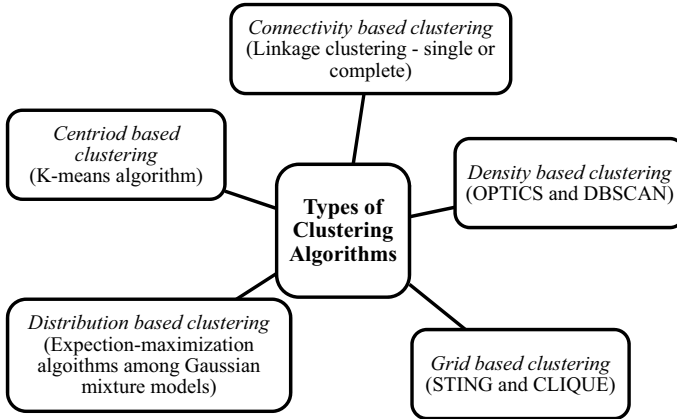
Fig. 9 Types of clustering algorithms

a cluster establishment and the process involving how to identify them competently. Hence it is required to study the differences between these algorithms on considering a classic cluster model that involves neural models, signed graph models, density models, group models and subspace models [24]. The classification of clustering are broadly as *hard clustering* (verifying all objects fits into a cluster or not) *and soft clustering* (verifying all objects fits into all the clusters to a definite level). It is further classified as hierarchical clustering (data of a child group also matches with parent), Strict portioning clustering with (each data matches with only one group) and without outliers (data matches with no groups), subspace clustering (overlying of data not possible within a subspace) and overlapping clustering (data matches with more than one group).

As stated above, clustering algorithms are classified according to the cluster models and the various types are depicted in Fig. 9. Connectivity-based clustering is according to the distance at which the objects are connected to form clusters more likely being linked to closer objects rather than the farther one. The distances are calculated by using the linkage principle. Clustering the objects with lesser distance is known as single-linkage clustering and clustering the objects with higher distance are known as complete linkage clustering. Centroid-based clustering is denoted by a principal vector called cluster vector that need not be associated with the data set. A specific approach called K-means clustering algorithm where a numerous amount of clusters are fixed to 'k' which is to be indicated in advance that provides a definite solution to the optimization problem by finding the k cluster and allot the objects to the nearby cluster center wherein it minimizes the square of distances from the cluster. A cluster model more relevant to statistics is done by distribution model based clustering that has the objects from the same distribution. It looks like how the artificial data sets are developed by selecting arbitrary objects from the distribution. The major issue in this method is overfitting and it is taken care by a method called Gaussian mixture models especially expectation minimization algorithm where modelling

of dataset is done by fixing the number of Gaussian distributions that are adjusted in random and the variables are optimized in iteration to have a best fit of data which will meet a local optima. Density-based clustering have clusters stated as parts of greater density than the rest of the data set. Objects lies in scarce areas which are needed to isolate clusters are taken as noise and margin points. The most familiar method of density-based clustering is DBSCAN which is same as linkage-based clustering where it is according to the distance of connecting points within the threshold conversely it relates the points that fulfill the density criteria stated as a lesser amount of further objects within that radius.

Another generalized method of DBSCAN is OPTICS which neglects the choice to select a suitable value for the range variable and develops a hierarchical output based on linkage clustering. Grid-based clustering algorithm is utilized for multifaceted data set that develops a grid-like structure and comparing the same by means of grids or cells. It is a quite quicker method with lesser complex in computing. It involves this sequence of operations: initially splits the data set into number of determinate cells, chooses a cell at random, and finding the density of that cell. If the cell's density is higher than the threshold then marking such cell as a new cluster, calculating the density of neighbor cells and if the neighbor cells are higher than the threshold then keep the cell in the cluster and this step is repeated until there are no neighbor cells with higher density than the threshold. This process is performed repeatedly till every cell is passing through.

The cluster analysis approach is applied extensively in the field of biology, bioinformatics, medical imaging, business and marketing, computer science, World Wide Web, social science, robotics, finance, petroleum geology, and lot more.

13 Deep Learning

Deep learning is one of the general techniques related to the artificial intelligence (AI) of supervised or unsupervised machine learning of data which is unstructured that mimics the handling of data by the human brain. Many of the deep learning models are built using artificial neural network (ANN). It is otherwise known as deep neural network. Deep learning is a course of machine learning algorithms that utilizes multiple layers in a network to predict correlation from the actual inputs with the target/output parameters that allow solving optimization problems in several practical applications. The architectures of deep learning shall be built layer by layer which aids to separate notions and choose the features that enhance the performance. Some of the architectures are deep belief networks (DBN), recurrent neural networks (RNN), convolutional neural networks (CNN) and deep neural networks (DNN). The term 'deep' addresses the number of layers that transform data from raw (Input) data to the target (Output) data using depth of credit assignment path (CAP) which defines the relation between the raw and target data [25]. For example, the CAP's depth in feedforward neural network is only one in addition to the number of hidden layers

whereas it is merely limitless in CNN since a signal may pass over a layer more than once [6].

Most of the deep learning algorithms are structured as problems of unsupervised learning where such algorithms utilize the unlabeled data rather than the supervising learning. The best example of an unsupervised trained deep structure is deep belief network. Figure 10 depicts the revolution of deep learning that shows by what means deep learning is a subdivision of machine learning and in turn it is a subdivision of AI. Since 2012 till date, deep learning in ANN has evolved widely from various works of different researchers as predicting target of bio-molecular drug, detection of deadly effects of environmental chemicals and household goods, recognition of image and object, computer vision, recognition of speech and classification of images using CNN and long short term memory (LSTM) methods [26, 27].

An ANN with several layers between the layers of input and output is called as deep neural network (DNN). Intricate non-linear relations can be modeled using DNN. Figure 11 shows the difference between number of layers in a typical feedforward ANN and DNN. The working of DNN is very similar to ANN which was described in detail in the earlier sessions except that DNN as 'n' number of hidden layers between the input and output layers. For example, in computer chess game, the learning of different moves or tactics can be learned by a computer from several people and the same can be stored in its database and those tactics are determined by various algorithms and that is why it can be termed as deep neural network where the learning is deeper where ANN is not an imaginative method where it can draw a single result while DNN will be able to solve the issues universally and can predict or conclude based on the input and the desired output. Similar to ANN, DNN also has two major issues of computational time and overfitting if it is not trained thoroughly.

Fig. 10 Revolution of Deep learning
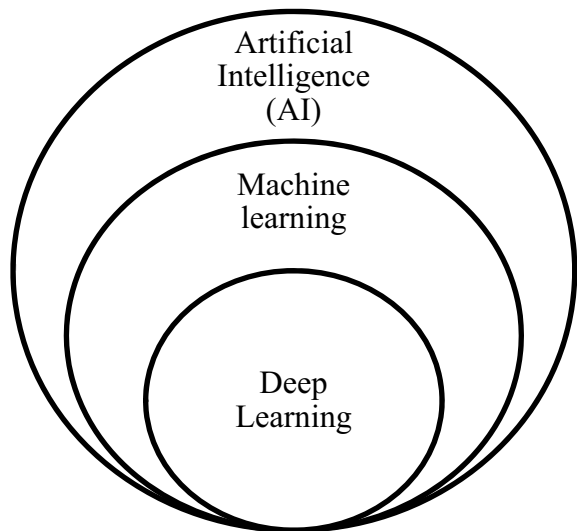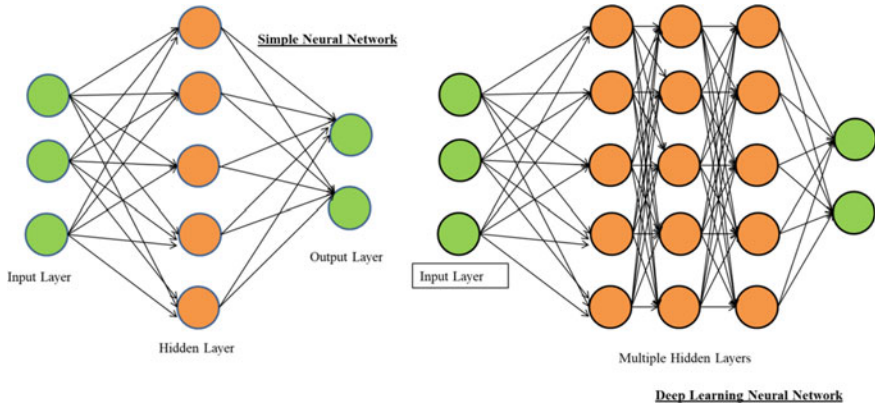
Fig. 11 A typical ANN and DNN architecture

The computation of several layers of a DNN with 'n' hidden layers is given by a relation as in Eq. (8)

$$f(x) = f\left[a^{(n+1)}(h^n(a^n(\ldots(h^2(a^2(h^1(a^1(x)))))\right]$$ (8)

$a^{(n)}(x)$ is the Pre-activation function shown in Eq. (9) that is a linear process with the weighted matrix $W^{(n)}$ and $b^{(n)}$ as bias which shall be merged to a variable θ

$$a^{(n)}(x) = W^{(n)}x + b^{(n)}$$ (9)

The bar notation $\bar{x}$d enotes that 'n' attached to the vector x and $h^{(l)}(x)$ is the hidden layer activation/transfer function and is given by Eqs. (10) and (11).

$$a^{(n)}(\bar{x}) = \theta^{(n)}\bar{x} \, if \, n = 1$$ (10)

$$a^{(n)}(\bar{h}^{(n-1)}) = \theta^{(n)}\bar{h}^{(n-1)} \, if \, n > 1$$ (11)

A neural network where the data shall direct in any ways is categorized as recurrent neural networks (RNNs), a division of ANN in which the nodal linkages create a focused graph beside a temporary arrangement that shows a temporary lively performance and RNN is utilized in an application like modelling of language. In specific, an active algorithm that is in use for this purpose is long short-term memory. A neural network that is being utilized in computer vision application for evaluating pictorial imagery is convolutional deep neural networks (CNNs) which basically depends on the shared architecture and features of constant translation and it is also used for the recognition of automatic speech by modeling good acoustics. CNNs are generally kinds of multilayer perceptrons that typically of fully connected networks in which every neuron in a single layer is attached to all the neurons in the consecutive layer.

This leads to a chance of overfitting of data that can be sorted out by including some form of weight measurements method to the functional loss. It is not that extreme due to its connectivity of complex patterns with tiny and easier patterns to various regularization approaches.

There are several applications in which deep learning concepts are utilized. They are drug discovery and toxicology, bioinformatics [28], customer relationship management, recognition of electromyography (EMG) and images, processing of natural language and visual arts, mobile advertising, military applications, and detection of financial frauds, etc. [6].

14   Summary

ML is a method of learning from the data based on the principles of statistics and AI. AI replicates human intellect in computers, like learning and critical thinking. Basically, AI deals with knowledge depiction and knowledge engineering. AI involves different types of approaches, like search algorithms, mathematical optimization, evolutionary algorithms, logic programming and automated reasoning, probabilistic methods for uncertain reasoning, classifiers and statistical learning methods.

Data analytics which practices business intelligence and analytics models is one of the major application areas of ML. Data analytics has four classes, viz. descriptive, diagnostic, predictive and prescriptive analytics. Descriptive analytics supports answering studies concerning the whereabouts. Diagnostic analytics supports answering inquiries regarding why things occurred. Predictive analytics supports answering inquiries concerning the later events. Prescriptive analytics supports answering inquiries regarding what has to be completed.

ML can be classified into three major classes. Supervised learning can be used if every sample in the data has an input item and a preferred output value. Unsupervised learning searches hidden configurations in a data set with no prior labels or outputs. Reinforcement learning movements of the software agents to exploit the best accumulative return.

Deep learning is the newest form of machine learning, which utilizes multiple layers in a network to select the features and to find the correlation among them to develop classification as well as predictive models of highly complex systems.

References

1. Davy Cielen, M. A., & Meysman, A. (2016). *Introducing data science: Big data, machine learning, and more, using python tools.* United States: Manning Publications.
2. Langley, P. (2011). The changing science of machine learning. *Machine Learning, 82*(3), 275–279.
3. Samek, W., Wiegand, T., & Müller, K. R. (2017). Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models, *1*, 39–48.

4. Shabbir, J., & Anwer, T. (2018). Artificial intelligence and its role in near future, *14*(8), 1–11.
5. Ginsberg, M. (2012). *Essentials of artificial intelligence.* San Francisco, CA, United States: Morgan Kaufmann Publishers Inc.
6. Dönmez, P. (2013). Introduction to machine learning. *Natural Language Engineering, 19*(2), 285–288.
7. Luger, W. (2004). George; stubblefield, *artificial intelligence: Structures and strategies for complex problem solving*, 5th ed. Benjamin/Cummings.
8. Makridakis, S. (2017). The forthcoming Artificial Intelligence (AI) revolution: Its impact on society and firms. *Futures, 90,* 46–60.
9. Johnston, J. (2010). *The allure of machinic life: cybernetics, artificial life, and the new AI*. Cambridge, Massachusetts London, England: The MIT Press.
10. Provost, R. K. F. (1998). Glossary of terms. *Machine Learning*, *30*. Springer US.
11. Le Roux, A., Bengio, N., & Fitzgibbon, N. (2012). Improving first and second-order methods by modeling uncertainty. In *Optimization for Machine Learning*, S. In Sra, Suvrit; Nowozin and S. J. Wright, Eds. MIT Press, 2012, p. 404.
12. Siegel, E. (2013). *Predictive analytics: The power to predict who will click, buy, lie, or die*, 1st ed. Wiley.
13. Hand, D. J., & Adams, N. M. (2015). *Data mining in Wiley StatsRef: Statistics reference online* (pp. 1–7). Chichester, UK: John Wiley & Sons Ltd.
14. Hilbert, M., & López, P. (2011). The world's technological capacity to store, communicate, and compute information. *Science (80), 332*(6025), 60–65, 2011.
15. Hashem, I. A. T., Yaqoob, I., Anuar, N. B., Mokhtar, S., Gani, A., & Ullah Khan, S. (2015). The rise of 'big data' on cloud computing: Review and open research issues. *Information systems, 47*, 98–115.
16. Barlow, H. B. (1989). Unsupervised learning. *Neural Computation, 1*(3), 295–311.
17. van Otterlo, M., & Wiering, M. (2012). Reinforcement learning and markov decision processes. In *Reinforcement Learning. Adaptation, Learning, and Optimization*, van O. M. Wiering M., Ed. Springer, Berlin, Heidelberg, pp. 3–42.
18. Nilsson, N. J. (2005). Introduction to Machine Learning—an early draft of a proposed textbook. *Machine Learning, 56*(2), 387–399.
19. Das, P., Bhattacharyay, B. K., & Datta, S. (2006). A comparative study for modeling of hot-rolled steel plate classification using a statistical approach and neural-net systems. *Materials and Manufacturing Processes, 21*(8), 747–755.
20. Mannila, H. (1996). Data mining: machine learning, statistics, and databases. In *Proceedings of 8th International Conference on Scientific and Statistical Data Base Management*, pp. 2–9.
21. Montgomery, G. G. V. D. C., & Peck, E. A. (2012). *Introduction to linear regression analysis*, 5th ed. Wiley.
22. Perry, S. W. (2002). Handbook of neural network signal processing. *Journal of the Acoustic Society of America, 111*(6), 2525–2526.
23. Prajapati, D. K., & Tiwari, M. (2017). Use of Artificial Neural Network (ANN) to determining surface parameters, friction and wear during pin-on-disc tribotesting. *Key Engineering Materials, 739,* 87–95.
24. Estivill-Castro, V. (2002). Why so many clustering algorithms. *ACM SIGKDD Explorations Newsletter, 4*(1), 65–75.
25. LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature, 521*(7553), 436–444.
26. Li, X. & Wu, X. (2015). Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4520–4524.
27. Sze, V., Member, S., Chen, Y., Member, S., & Yang, T., Efficient Processing of deep neural networks: A tutorial and survey, pp. 1–32.
28. Choi, E., Schuetz, A., Stewart, W. F., & Sun, J. (2017). Using recurrent neural network models for early detection of heart failure onset. *Journal of the American Medical Informatics Association, 24*(2), 361–370.

Neural Network Model Identification Studies to Predict Residual Stress of a Steel Plate Based on a Non-destructive Barkhausen Noise Measurement

Tero Vuolio, Olli Pesonen, Aki Sorsa, and Suvi Santa-aho

Abstract In this study, a systematic comparative study is made between selected ways for identification of a neural network model for the prediction of the residual stress of steel based on the non-destructive Barkhausen noise measurement. The compared approaches are the deterministic forward selection with and without filter and a stochastic genetic algorithm-based approach. All the algorithms make use of the extreme learning machine as a model basis. The main objective is to propose a systematic procedure for identifying a prediction model for the considered system. The results of this study show that the algorithmic approach might be considered necessary not only to reduce the effort for model selection but also to select models with high prediction performance. It was also found that the genetic algorithm proposed earlier by the authors is applicable for selecting a well-generalizing model to the system, but the performance of the deterministic selection techniques is also comparable to the genetic algorithm.

T. Vuolio (✉)
Process Metallurgy Research Unit, University of Oulu, P.O. Box 4300, 90014 Oulu, FI, Finland
e-mail: tero.vuolio@oulu.fi

O. Pesonen
Environmental and Chemical Engineering, University of Oulu, P.O. Box 4300, 90014 Oulu, FI, Finland

A. Sorsa
Department of Process and Environmental Engineering, University of Oulu, P.O. Box 4300, 90014 Oulu, FI, Finland

S. Santa-aho
Materials Science and Environmental Engineering, University of Tampere, P.O. Box 589, 33014 Tampere, Finland

29

1 Introduction

Residual stress is an important factor to evaluate in a manufactured steel plate, as tensile stress may cause an unexpected failure under an external stress while compressive stress improves fatigue resistance. Barkhausen noise measurement is an interesting method for residual stress evaluation because the measurement is non-destructive and fast and thus suits online applications [17] This, however, requires models that link Barkhausen noise signal characteristics to material property of interest. However, the identification of such a dependency is not straightforward because Barkhausen noise has been observed to be dependent on various material properties and residual stress [4]. Effects of these factors cumulate to the measured signal obscuring the effect of a single factor that is of interest. Furthermore, the dependencies are complex and non-linear. Barkhausen noise is also a stochastic phenomenon and because of that only averaged properties of the measured signals are reproducible. Depending on the number of measurement repetitions, uncertainty may be significant when applying Barkhausen noise measurement [26].

Predictive models between Barkhausen noise and residual stress can be found in the literature. Some examples are as follow: A multiple linear regression model was identified in Sorsa et al. [23] for residual stress and hardness. Features for the models were selected with a forward-selection algorithm. A similar study was carried out in Sorsa et al. [24]. The features were selected with a genetic algorithm preceded by a successive projection algorithm, decreasing the dimensionality of the data. A PLSR model was applied in Sorsa et al. [25] to predict residual stress from BN measurements. PLSR model identification was preceded by feature elimination. Feature elimination leads to improved results, especially with testing data. Ghanei et al. [7] found that a second-order polynomial described the relationship between BN and material properties. In Sorsa et al. [29], a non-linear regression model was fitted into the data from nitrided samples to predict residual stress. Feature selection was carried out with a backward elimination algorithm. Wang et al. [28] applied artificial neural networks to predict residual stress. They selected the input features manually. Ghanei et al. [8] used an adaptive neuro-fuzzy system with manually selected features to predict material properties based on BN measurements. Also, a theoretical model is derived between BN and residuals stress [15]. The model describes the frequency of BN and shows how it is affected by residual stress. The theoretical model needs a measurement of magnetic permeability.

The data sets obtained from the Barkhausen noise signal are multivariate, non-linear, collinear, and usually contain a limited number of observations. The non-linearity rises from the complexity of the phenomenon, and thus many feature candidates are usually extracted from the signal, making the data multivariate. These characteristics set high requirements especially for the model identification and validation stages. It is also well known that the productional environment itself, changing material properties and measurement uncertainty introduce variation in the data, which complicates the situation a bit and must be considered.

2 Model Identification

In this study, the modeling scheme is divided into five consecutive steps, which are (1) feature extraction, (2) feature selection, (3) model structure selection, (4) final model training and (5) model performance assessment (Hastie et al. [12]. The model selection flowchart is presented in Fig. 1. The next sections and subsections provide the details of these steps. The prediction model outcome can be presented in a general form as [12]:

$$E(y|X, \theta) = f(X, \theta), \tag{1}$$

where E(y | X) is the expected value for the output variable given the input set X, $f(X, \theta)$ is the functional form of the model, θ is the model parameter vector. The prediction model identification simply means the selection of a model that predicts system outcomes with adequate accuracy. In the case of regression problems, the outcome to be predicted is defined in the continuous space. As the prediction models are identified in such a way that the prediction error is minimized, the identification procedure itself can be considered to fall into the category of supervised learning. In supervised learning, the best model is chosen based on the derived objective function. In regression problems, the usual criteria are based on a squared prediction error.

The objective function for the model identification can be considered multiobjective, as it follows the principles of Occam's Razor, meaning that the best model is the one that predicts the behavior of the system with the least complexity. This principle is strongly connected to the well-known concept of bias-variance trade-off. This trade-off can be simplified with the following intuitive rule-base:
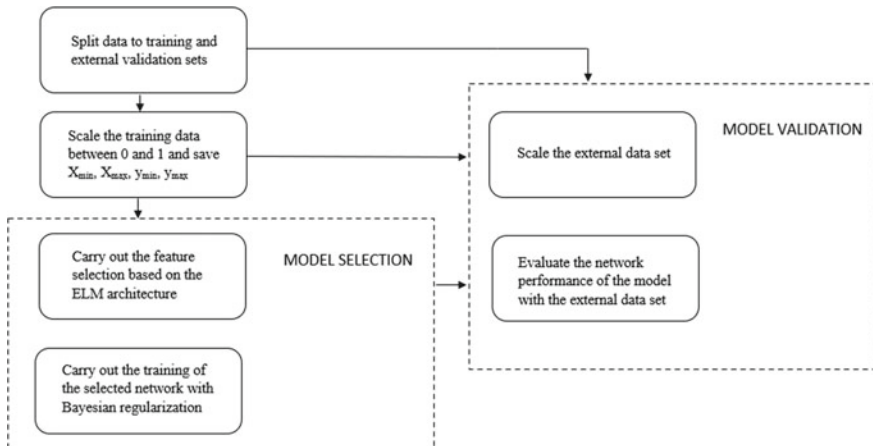
Fig. 1 The model selection flowchart

(1) The models that have a large prediction error and low complexity, have high bias, but low variance.
(2) The models that have a small prediction error, but a high complexity, have a small bias, but low variance.

However, the prediction performance of a model is rather a matter of generalizability than of minimizing the error for the training; in order for a model to be generalizable, it has to perform well on an out-of-sample data set, which, in this study, is referred to as the external validation set. To improve the generalizability, numerous alternatives have been proposed for model selection and are further discussed, for example, in the following references [1, 16]; Schwenk [19, 22]. In this study, the applied methods are based on cross-validation and model regularization.

2.1 Feature Extraction

Barkhausen noise as itself is quite useless and thus for prediction, usually a set of features are computed that represent signal characteristics. The prediction models are then identified between these features and the material property of interest. Traditional features are the RMS value (root mean square) and height, width and position of the so-called Barkhausen noise profile. However, several other features can be used such as different statistical values, hysteresis loop parameters, factors, entropies, generalized norms and so on. Furthermore, features are usually computed from the time-domain signal while feature extraction can also be extended to frequency-domain. Obviously, in the feature extraction stage, the number of feature candidates easily shoots up and collinearity problems may arise. Because of this, the feature selection needs to be carried out [9, 10].

2.2 Feature Selection

The feature selection is often considered as the most computationally demanding step in model identification [9]. Feature selection is carried out primarily to reduce model complexity and thus improves the generalizability. However, an improved interpretability of the models is achieved with feature selection, the feature selection methods can be categorized in various ways, the main division is usually made between the manual and automatic feature selection. The first one of these considers the use of expert knowledge in the selection, while the latter introduces the use of algorithmic approaches. What comes to feature selection algorithms, these can be further divided into wrappers and filters and further into deterministic and stochastic approaches. The filter approaches are based on model-free feature ranking, and the wrappers estimate the usefulness of the features based on the objective function [9]. The main difference between the deterministic and stochastic approaches is that the

deterministic algorithms end up to the same solution with the same initial conditions, as the stochastic approaches do not.

Referring to the bias-variance trade-off, the feature selection step is crucial. To find this trade-off, this study uses the sum of squared error estimated with repeated random subsampling method in the inner validation loop as the objective function. In the repeated random subsampling method, the data are repeatedly split in training and internal validation sets without replacement, and the modeling error is assessed based on the specified error function for each of the split repetitions. The error estimate for the model is taken as the average over these cross-validation repetitions. In other words, the objective function is as follows:

$$\min J = \min \frac{1}{N} \sum_{j=1}^{N} e(y_{cv,j}, \hat{y}_{cv,j}), \tag{2}$$

where J is the objective function, N is the cross-validation repetitions, e is the specified error function, y_{cv} is the observed residual stress for the internal validation set and $\hat{y}_{cv}$ is the predicted outcome for the internal validation set. In this study, the used error function is the sum of squared error averaged over N split repetitions. If the cross-validation is not used, then the error is computed based on the training set only.

2.2.1 Manual Selection

If the input variables for the model are selected by making use of domain knowledge, it is often referred to as the manual selection in the literature. Burnham and Anderson [2] suggest that the manual selection should be preferred as the selection strategy, as the domain knowledge contains valuable information of the system that the data do not easily reveal [2]. In this study, the work of Santa-aho et al. [21] and Sorsa et al. [24] was used to extract the domain knowledge. In this study, the manual selection is used to select the final model based on the statistics of the repeated tests carried out with the search algorithms.

2.2.2 Greedy Forward Selection (FS)

The greedy forward selection algorithm is a feature selection technique, which starts from a feature vector full of zeros, i.e. from an empty model. During the search, the features are added into the model in a greedy manner, meaning that for each iteration, the updated model that yields the best result is selected as the starting state for the next iteration. This procedure is continued until a specified stopping criterion is met. In this study, the stopping criterion is defined as the explicit improvement of the model performance. The main problem of the forward selection, as well as

Table 1 The forward selection algorithm used in this study

The Forward selection

```
While J_{k+1} < J_k

      For each x_i in X_k = {1, 2 … M-k}
            1) Add feature candidate x_i to the trial model
            2) Evaluate the objective function for the trial model
      end

   1) Select the best model along the set of M-k trial models
   2) If the objective function value is improved, update the current model
      containing k number of features

end
```

Notes M = Total number of features in the set, k = Number of features in the current model

the other deterministic approaches, is that it tends to stick in a local optimum. This could mean for example that the model that is selected with the algorithm does not contain all the relevant features or, on the other hand, contains a significant number of redundant features. In addition, the number of hidden neurons could be too low or high to estimate the relation between the inputs and the output, potentially resulting as under or overfitting. The pseudocode for the selection algorithm is given in Table 1. The proposed approach can be also found to be called in the literature as the hill-climbing algorithm. [14]

2.2.3   Forward Selection with Ranking

In this approach, prior to selection, the features are ranked based on the absolute value of their correlation coefficient in descending order. After the ranking, all the features are sequentially added or discarded from the model, depending on whether the addition of the feature candidate improves the objective function value. The main purpose of this approach is to reduce the computational complexity of the traditional forward selection. However, the approach suffers from the same limitations as does the algorithm presented above.

2.2.4   Genetic Algorithm

In this study, genetic algorithms are used as a stochastic alternative for the deterministic engines. The genetic algorithm has been proven effective for solving both

constrained and non-constrained optimization problems. The genetic algorithm solution space constitutes of n chromosome vectors, each of which represents a solution to the problem. The genetic algorithm is based on the principles of Darwin's natural selection, as it uses simplified genetic operations such as selection, crossover, and mutation for improving the solution vectors. The number of genes in the chromosome vector corresponds to the overall number of features in the set. In a genetic algorithm-based feature selection, each chromosome corresponds to a model candidate (or a trial model), encoded as a binary vector. In this representation, the gene values are defined as x = {0,1}, meaning that if the value of a gene in the feature vector is 1, the corresponding feature is included in the model. It has been demonstrated by Deniz and Kiziloz [5] that the initialization of the genetic algorithm population affects the selection performance, and a pure random initialization often performs badly in the case of data sets with a large number of features. Thus, in this study, the population is initialized such that 50% of the population is initialized pure randomly, and in the latter 50%, the number of ones is 10 at maximum. This way, the search can be guided toward the selection of more parsimonious models. There are numerous ways to implement the genetic algorithm, of which this study uses the following operators:

(1) Roulette-wheel selection,
(2) Crossover with two cut-off points,
(3) Dot mutation and a segment mutation,
(4) Elitism for a single individual.

The population size as well as the rate of crossover and mutation is a very important hyperparameter for convergence of the algorithm. In this study, the mutation probability evolves deterministically within the generations, and the cross-over probability is set as constant $p_C = 0.9$. The other hyperparameters, i.e. the number of individuals and the properties of the objective function are discussed in more detail in the Results and Discussion section. A more detailed description of the algorithm and the evaluation of its performance in neural network model identification problems can be found [27].

2.3　Neural Network Structure Selection

The Extreme Learning Machine (ELM) is used as the model basis in the feature selection phase. This because the computational complexity of ELM training is very small compared to backpropagation algorithms [13]. In the deterministic approaches, the number of hidden neurons was selected with the grid search, which is practically the only way to implement the selection of hidden neurons within the deterministic search, as the algorithm requires the model structure that is to be evaluated. This issue increases drastically the time complexity of the algorithms, as the grid search demands an extra loop to the implementation. In the case of the genetic algorithm, there are some alternatives for the selection of hidden neurons, including grid search, ranking the models with different numbers of hidden neurons before the variable

selection [3] or just carrying out the search with constant number of hidden neurons [5]. It is also possible to include and encode the number of hidden neurons in the individuals. To reduce the computational burden, the number of hidden neurons was expressed as a single integer value. By this, every individual is essentially a fully connected network with a certain number of hidden neurons. Thus, the encoding of the genetic algorithm is a hybridization of binary and integer-based coding, where the binary part encodes the feature vector and the integer part encodes the network structure [27]. The integer-based coding makes use of simulated binary cross-over [20] and Mäkinen–Periaux–Toivonen mutation [18].

2.4   Final Model Training

The final model training is carried out with the Bayesian regularization algorithm, which has been proven effective for improving the generalizability of the neural network models for small data sets. In the regularization, the variance of the model parameters is minimized simultaneously with the prediction error, resulting in more stable models, which thus generalize better to out of sample data sets. The detailed description of the algorithm is provided in Foresee and Hagan [6].

2.5   Model Performance Assessment

The model performance was assessed by making use of coefficient of determination (R^2) and mean absolute error (MAE). The details for these computing the metrics can be found, for example, in Harrell [11]. The tendency for overfitting was evaluated simply by computing the corresponding metrics for training and test set.

3   Considered Data Set

The Barkhausen noise measurements were carried out for a set of samples that were machined from RAEX400 low alloyed, cold-rolled steel. The samples were carburizing case-hardened and then tempered with different tempering temperatures and durations to obtain different surface hardness conditions. The samples were then subjected to a bending load to achieve different stress states to sample surfaces. Altogether 98 data points including Barkhausen noise and residual stress measurements were recorded from the samples. A more thorough description of samples, their preparation and measurements can be found in Santa-aho et al. [21].

After the feature extraction from the Barkhausen noise signal, the overall data set contains 48 feature candidates, which makes the number of model candidates extremely high compared with the number of data points. To complicate the search,

the number of feature candidates was doubled such that the rows of the exact replicate of the feature set were randomized and attached within the original set, making the overall number of features 96. Consequently, as the complexity of the exhaustive search is $n_{HN,max} \cdot N \cdot (2^n - 1)$ with the proposed objective function, meaning that for an explicit global optimum, $7.92 \cdot 10^{28} \cdot n_{HN,max}$ models would have to be evaluated, each for N data subset repetitions in the internal validation loop, keeping in mind that there are $n_{HN,max}$ hidden neurons in the search grid, based on which the neural network structure is selected. It is obvious that this high number of models is not possible to be evaluated in a feasible amount of time. In general, the excessively number of available models highlights the need for feature selection algorithms in the case of industrial data sets.

4 Results and Discussion

To comparatively study the performance of the model identification algorithms, an experiment was carried out. The deterministic algorithms were tested with varying objective function parameters, i.e. numbers of internal validation loop splits ($N =$ [0 1 10 20 50]), fraction of data in the internal validation ($l =$[0 0.2 0.35 0.5]) and maximum number of hidden neurons in the grid search ($Num_{HN,max} =$ [1 5 12 20]). This results in 49 experimental design levels for the deterministic algorithms. Because of the computational complexity and the vast number of other hyperparameters, the design variables for the genetic algorithm were reduced to two ($l =$[0.35 0.5] and N = [1 20 50]). The number of hidden neurons was set to $Num_{HN,max} = 20$, i.e. the initial population contained individuals with 1–20 neurons. The number of individuals in the population was $n_{pop} = 200$. The maximum number of generations was set to $gen_{max} = 100$. This is mainly because it has been found in the earlier study of the present authors [27] that the use of cross-validation as the objective function improves drastically the performance of the algorithm, and for a large number of feature candidates, a sufficiently large population is beneficial to be used.

Prior to experiments, the external validation set was extracted from the data. The split was made with stratification. To cover the whole measurement range for the test set, it was selected by sorting the data in descending order with respect to measured residual stress, after which every seventh sample was chosen for external validation. Each of the algorithms was run 10 times with different parameters, and the details of the models and the performance metrics were systematically recorded. To evaluate the model selection performance of the algorithms, a simple discretized performance scale was proposed. The performance scale was determined by evaluating the usability of the model in practical applications. It should be noted that the scale was used for the final models only, i.e. to models trained with Bayesian Regularization. The performance scale is presented in Table 2.

Table 2 Discretized
evaluation criteria of the
model performance

MAE$_{test}$ [MPa]	Model performance category
<50	Excellent
[50, 80]	Decent
>80	Bad

4.1 The Selected Features

The greedy forward selection selected more parsimonious models on average (3.6 features), as the forward selection with ranking selected 4.1 features on average over 49 experiments, with 10 repetitions for each. In Fig. 2, the histograms of the selected number of features are presented. As seen in the figure, the mode of the greedy search has two features, whereas, for the forward selection with filter, the mode has four selected features. Among all excellent models, the most common features occurring in the deterministic search results were signal-to-noise ratio (86%), power spectrum density (53%), entropy of the signal (36%) standard deviation of the signal (27%) and root-mean-square of the signal (18%).

Neither of the algorithms selected consistently the randomized features, even though the forward selection with filter selected these occasionally. Figure 3 presents the selected features for the deterministic algorithms for all the test repetitions, from which it can be seen that the random features (features 48–94) are in fact selected very rarely.

On average, the genetic algorithm selected 6.6 features over 60 repetitions (six experiments with 10 repetitions for each). In rare cases (five cases, i.e. 8%), the genetic algorithm diverged to a relatively large subset (~30–50 features). The histogram of the selected number of features with the genetic algorithm is presented in Fig. 4. In the case of genetic algorithm, the top five features were the signal-to-noise
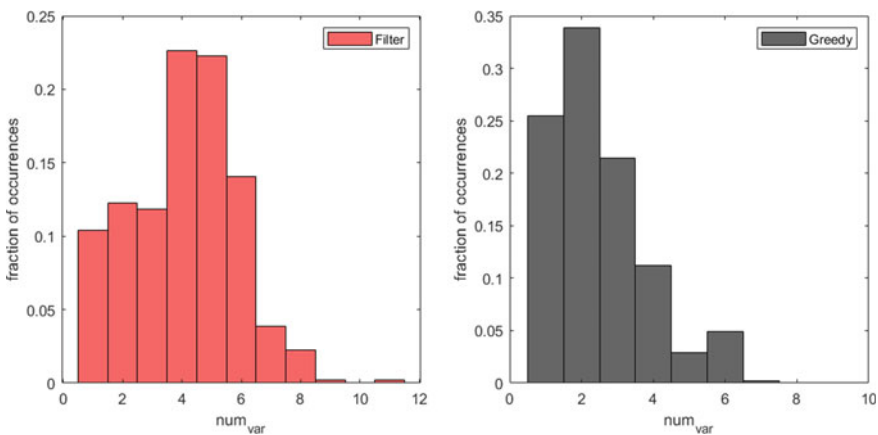
Fig. 2 Histograms for the number of selected features acquired with the deterministic algorithms
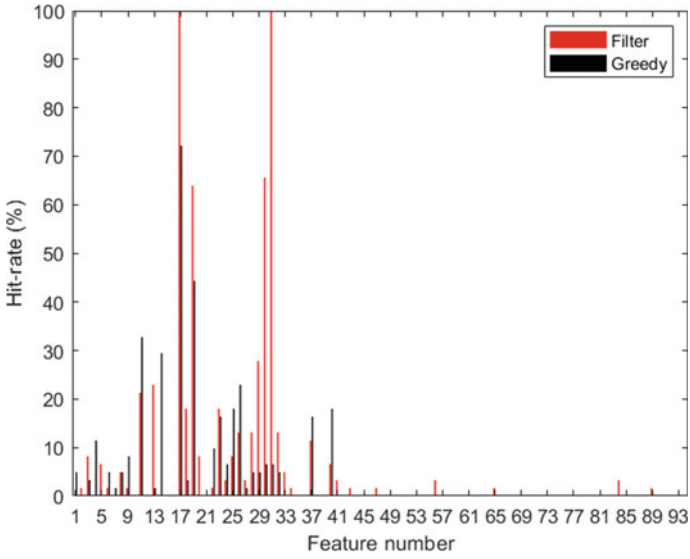
Fig. 3 Occurrence of the features for all the experiments for the deterministic algorithms. Hit-rate (%) is the percentage of the occurrences with respect to all test repetitions
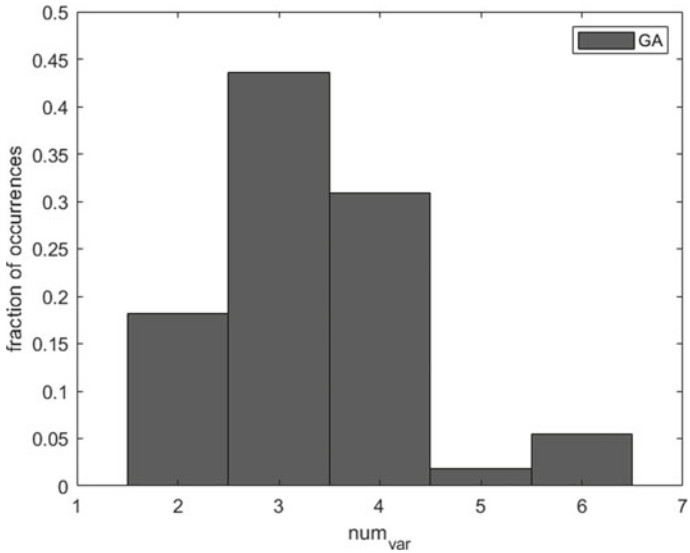
Fig. 4 Histograms for the number of selected features acquired with the genetic algorithm. The histogram is counted without the diverged cases

ratio (92.9%), spectrum area (35.7%), root-mean-square of the signal (35.7%) area of the hysteresis loop (28.6%), standard deviation of the signal (28.6%). However, 60% of these occurred when the number of split repetitions was set to $N = 1$. Thus, the risk of divergence could be reduced with increasing cross-validation repetitions, to avoid a badly generalizing subset to be highly ranked just by chance. In addition, the risk of divergence to large feature sets can be decreased by initializing the population to contain a small number of ones and letting the algorithm to converge to a single solution, and by that avoiding a pre-mature convergence. However, this increases the computational time of the algorithm significantly for the cases where the initial population is very far from the optimum. Further, it was found that the consistency of the search could be improved with cross-validation, which, however, increases the computational load of the algorithm. The selected features are only partially in agreement with the earlier studies, Sorsa et al. [23]. However, Sorsa et al. [23] used multiple linear regression as the model basis, which is not straightforward to be compared with neural networks due to different model structures.

All and all, 50% of the models identified in all experiments with the deterministic algorithms were at least decent (MAE$_{test}$ < 80 MPa), whereas from the models found with genetic algorithm, 70% were considered at least decent. However, it should be noted that the experimental design was non-similar for the stochastic and the deterministic algorithms. For the same computational parameters, i.e. number of cross-validation repetitions and fraction of data for internal validation, 68% of the models identified with forward selection with ranking were at least decent, whereas, for greedy forward selection, the corresponding percentage was 61%.

Further analysis of the experiments carried out with deterministic algorithms shows that both algorithms select excellent models for 12% of all cases. For a forward selection with ranking, if a hold-out or no cross-validation was used as the objective function, 9.9% of the models were excellent. If a repeated cross-validation was used instead, 13.4% of the models were excellent. Similarly, for greedy forward selection, 9.9% (hold-out or no cross-validation) and 14.4% (repeated random subsampling) of the models were excellent.

Further, a logistic regression model was constructed to analyze the effect of computational hyperparameters on the risk of selecting an excellent model. It was found for both algorithms only the number of hidden neurons had a statistically significant effect on the model performance class, when a risk level of $\alpha = 0.05$ was used in hypothesis testing. In fact, if the search results with the maximum number of hidden neurons below 20 are filtered out, the percentage of excellent models is around 19% for the forward selection with ranking and 17% for greedy forward selection. This indicates that the risk of underfitting can be decreased by widening the search grid with respect to a number of hidden neurons. Along with the number of hidden neurons, the training method had a significant impact on the model performance; if the hidden layer weights were left untrained (the model is the extreme learning machine), only 0.7–3.5% of the models were excellent. However, the computational parameters had an effect on the convergence of the genetic algorithm; similarly, as in our previous work [27], the use of cross-validation decreased the standard deviation of the selected features. In Fig. 5, the effect of the number of cross-validation repe-
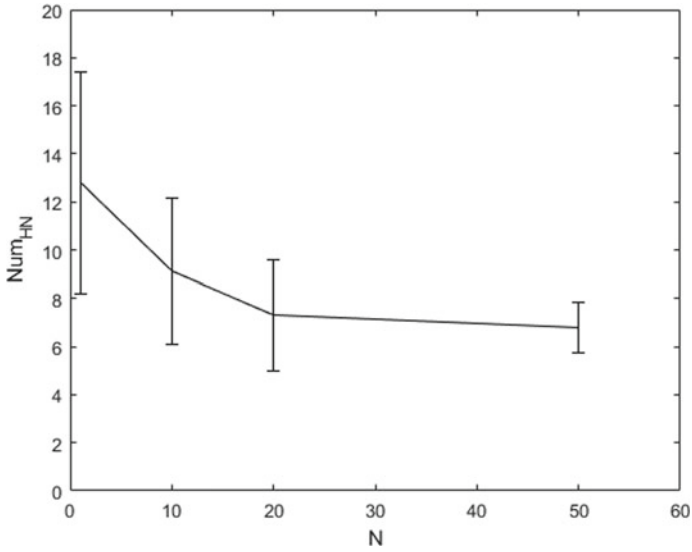
Fig. 5 Selected number of hidden neurons in the final model as a function of cross-validation repetitions in the inner validation loop. The results are obtained with Forward Selection with ranking

titions to the number of selected neurons in the best models is illustrated in the case of forward selection with ranking. The fraction of data held out in the internal validation error estimation is set to $l = 0.5$ and the maximum number of hidden neurons is $\text{Num}_{\text{HN,max}} = 20$. It is seen in the figure that the number of hidden neurons as well as the standard deviation between the repetitions decreases with the number of cross-validation repetitions. This result partially indicates that the increased number of cross-validation repetitions increases the repeatability of the search, provided that the fraction of the data held out in the split is chosen with care.

The illustration of the bias-variance trade-off is presented in Fig. 6. From the figure, it is clearly seen that as the model complexity increases with respect to network width, the error for the internal validation set does not follow the error obtained for the training set and starts to increase if there are over 10 neurons in the model. This behavior is an explicit proof of overfitting of the network. This matter highlights the need for internal cross-validation during the feature selection phase, as the training error is clearly a highly optimistic estimate of the hypothetical performance of the model error for the external validation set, even though the risk associated to selecting excellent models was not found dependent on the computational parameters.
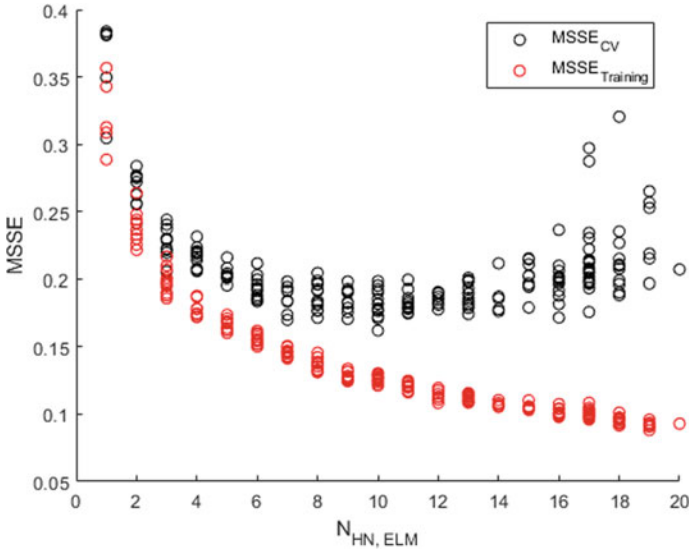
Fig. 6 The objective function values for training and internal validation sets for the final population of the genetic algorithm. The error estimates are computed for the Extreme Learning Machine

4.2 Final Model Selection and Model Performance Assessment

The final model was chosen such that the features that appeared most frequently among the set of excellent models were chosen for the final model. Consequently, the final model study constituted of a set of five features (hysteresis loop area, spectrum area, standard deviation of the signal, signal-to-noise-ratio and root-mean-square) and 11 hidden neurons. The corresponding figures of merit were $R^2 = 0.97$ and MAE = 36.8 MPa for the test data. In Fig. 7, the predicted residual stresses for the steel plate are presented. As seen from the distribution of the predictions for the external validation set, the generalizability of the model can be considered very good. In other words, the prediction results for external validation and the training set were consistent, and thus the model is expected to behave consistently also for hypothetical previously unseen data. However, the model selection as well as the final calibration model would obviously benefit from more data. Still, the results of this study seem convincing.
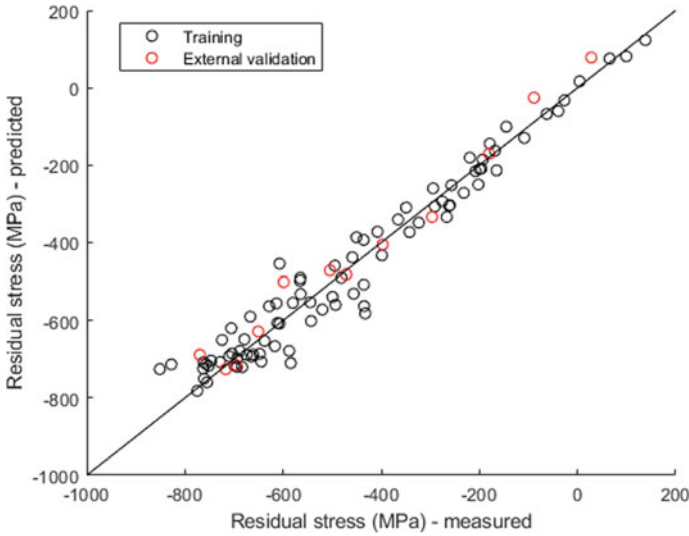
Fig. 7 Measured and predicted residual stress of a steel plate. The exemplified model is identified based on the results of the Genetic Algorithm

5 Concluding Remarks and Recommendations

In this study, three model selection algorithms implemented to be suited for neural network model selection were applied to predict a residual stress of a steel plate based on a non-destructive Barkhausen noise measurement. The results of this study show that the model identification algorithms can assist the modeler in model selection task, when solving industrially relevant regression problems even for small data sets. Regardless of the algorithm in use, systematic tuning of the computational parameters is needed to obtain reasonable results. It is shown that all the algorithms can select a relatively good set of features to predict a residual stress of a steel plate, but if the excellency criterion is set to MAE < 50 MPa, it is quite rare that the excellent models are found. However, if the criterion for an applicable model is set to decent (with the similar discretization as in this study), around 60–72% of the identified models regardless of the algorithm are applicable under conditions of this study. It should be noted that the proper discretization criterion is heavily dependent on the final application as well as on the experimental data. It was also noticed that the use of cross-validation as the objective function in model selection is beneficial regardless of the computational complexity, as it is obvious that the training error is an optimistic estimate of the model prediction performance. However, the cross-validation parameters, namely the number of split repetitions and the fraction of data held out in the split must be chosen with care. It was observed that by increasing the number of splits, the consistency of the selection could be improved for the deterministic

algorithms. Otherwise, there is a risk for coincidental selection. Thus, the performance of the model selection algorithms can be considered at least reasonable, but the importance of the domain knowledge should not be underestimated. In future work, it is obvious that the performance of the models would benefit from more data.

References

1. Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control, 19*(6), 716–723.
2. Burnham, D. R., & Anderson, K. P. (2002). *Model selection and multimodel inference: A practical information-theoretic approach.* New York: Springer.
3. Chyzhyk, D., Savio, A., & Grana, M. (2014). Evolutionary ELM wrapper feature selection for Alzheimer's disease CAD on anatomical brain MRI. *Neurocomputing, 2014,* 73–80.
4. Davut, K., & Gür, G. (2007). Monitoring the microstructural changes during tempering of quenched SAE 5140 steel by Magnetic Barkhausen noise. *Journal of Nondestructive Evaluation, 26,* 107–113.
5. Deniz, A., & Kiziloz, H. (2019). On initial population generation in feature subset selection. *Expert Systems with Applications, 137,* 11–21.
6. Foresee, F., & Hagan, M. (1997). Gauss-Newton Approximation to Bayesian learning. *Proceedings of International Joint Conference on Neural Networks*, pp. 1930–1935.
7. Ghanei, S., Saheb Alam, A., Kashefi, M., & Mazinani, M. (2014). Nondestructive characterization of microstructure and mechanical properties of intercritically annealed dual-phase steel by magnetic Barkhausen noise technique. *Materials Science and Engineering A, 607,* 253–260.
8. Ghanei, S., Vafaeenezhad, H., Kashefi, M., Eivani, A. R., & Mazinani, M. (2015). Design of an expert system based on neuro-fuzzy inference analyzer for on-line microstructural characterization using magnetic NDT. *Journal of Magnetism and Magnetic Materials, 379,* 131–136.
9. Guyon, I., & Elisseeff, A. (2003). An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research, 3,* 1157–1182.
10. Guyon, I., & Elisseeff, A. (2006). An introduction to feature extraction. In *Feature extraction* (pp. 1–25). Springer, Berlin, Heidelberg.
11. Harrell, F. E. (2015). *Regression modeling strategies: With applications to linear models, logistic and ordinal regression, and survival analysis.* Springer.
12. Hastie, T., Tibshirani, R., & Friedman, J. (2017). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (2017), Springer Series in Statistics.
13. Huang, G. B., Zhu, Q. Y., & Siew, C. K. (2006). Extreme learning machine: Theory and applications. *Neurocomputing, 70*(1–3), 489–501.
14. Kohavi, R., & John, G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence, 97*(1–2), 273–324.
15. Kypris, O., Nlebedin, I. C., & Jiles, D. C. (2014). A model for the Barkhausen frequency spectrum as a function of applied stress. *Journal of Applied Physics, 115*, 083906.
16. Mallows, C. L. (2000). Some comments on Cp. *Technometrics, 42*(1), 87–94.
17. Moorthy, V., Shaw, B., Mountford, P., & Hopkins, P. (2005). Magnetic Barkhausen emission technique for evaluation of residual stress alteration by grinding in case-carburised En36 steel. *Acta Materialia, 53,* 4997–5006.
18. Mäkinen, R., Periaux, J., & Toivanen, J. (1999). Multidisclipnary shape optimization in aerodynamics and electromagnetics using genetic algorithms. *International Journal for Numerical Methods in Fluids, 30,* 149–159.
19. Nowak, R. D. (1997). Optimal signal estimation using cross-validation. *IEEE Signal Processing Letters, 4*(1), 23–25.

20. Ripon, K. S. N., Kwong, S., & Man, K. F. (2007). A real-coding jumping gene genetic algorithm (RJGGA) for multiobjective optimization. *Information Sciences, 177,* 632–654.
21. Santa-Aho, S., Vippola, M., Saarinen, T., Isakov, M., Sorsa, A., Lindgren, M., et al. (2012). *Barkhausen noise characterisation during elastic bending and tensile-compression loading of case-hardened and tempered samples, 47,* 6420–6428.
22. Schwenk, H., & Bengio, Y. (2000). Boosting neural networks. *Neural Computation, 12*(8), 1869–1887.
23. Sorsa, A., Leiviskä, K., Santa-aho, S., & Lepistö, T. (2012). Quantitative prediction of residual stress and hardness in case-hardened steel based on the Barkhausen noise measurement. *NDT and E International, 46,* 100–106.
24. Sorsa, A., Leiviskä, K., Santa-aho, S., Vippola, M., & Lepistö, T. (2013). An efficient procedure for identifying the prediction model between residual stress and Barkhausen noise. *Journal of Nondestructive Evaluation, 32*(4), 341–349.
25. Sorsa, A., Isokangas, A., Santa-aho, S., Vippola, M., Lepistö, T., & Leiviskä, K. (2014). Prediction of residual stresses using partial least squares regression on Barkhausen noise signals. *Journal of Nondestructive Evaluation, 33*(1), 43–50.
26. Tomkowski, R., Sorsa, A., Santa-Aho, S., Lundin, P. & Vippola, M. (2019). *Statistical evaluation of barkhausen noise testing (BNT) for ground samples,* Sensors 19, Article number 4717.
27. Vuolio, T., Visuri, V.-V., Sorsa, A., Ollila, S., & Fabritius, T. (2020). Application of a genetic algorithm based model selection algorithm for identification of carbide-based hot metal desulfurization. *Applied Soft Computing Journal*, *92*, Article number 106330.
28. Wang, P., Zhu, L., Zhu, Q., Ji, X., Wanga, H., Tian, G., et al. (2013). An application of back-propagation neural network for the steel stress detection based on Barkhausen noise theory. *NDT and E International, 55,* 9–14.
29. Sorsa, A., Santa-aho, S., Aylott, C., Shaw, B. A., Vippola, M., & Leiviskä, K. (2019). Case Depth Prediction of Nitrided Samples with Barkhausen Noise Measurement. *Metals, 9*(3), 325.

Data-Driven Optimization of Blast Furnace Iron Making Process Using Evolutionary Deep Learning

Bashista Kumar Mahanta, Rajesh Jha, and Nirupam Chakraborti

Abstract Optimization techniques are applied widely in iron and steel making process to solve complicated process-related problems. These methods and the models created through them are regularly used in this field to find out the optimum operating conditions in terms of cost, quality, quantity and effectiveness of the process. The various blast furnace parameters like burden distribution, oxygen enrichment, productivity improvement, composition of the top gas, quality of hot metal production, etc., are very difficult to effectively optimize. In recent times data-driven models of diverse nature have been quite successfully applied for this purpose, where evolutionary approaches have made a significant impact in simultaneous optimization of multiple number of objectives in problems related to the iron and steel making industry. In this chapter implementation of various evolutionary strategies are discussed, which are recently applied in this domain to tackle some real-world problems.

1 Background

Global steel demand is increasing day by day. To fulfill the demand blast furnace iron making route is used in every part of the world. This route is used to produce about 94% of the total iron consumed by the steel industry. Large numbers of technological innovations and design changes have been made for the blast furnace operation to improve the quality, rate of production, and the reduction of process cost. To meet the global aspiration, optimization acts as an important tool in the blast furnace operation. This helps to produce good quality steel through an improved operational strategy [1, 2]. When the process begins or some changes in it are envisaged, the optimization

B. K. Mahanta · N. Chakraborti (✉)
Department of Metallurgical and Materials Engineering, Indian Institute of Technology, Kharagpur, India

R. Jha
Department of Mechanical and Materials Engineering, Florida International University, Miami, FL, USA

and data mining methods can be applied to the system in order to determine the optimized parameters for smooth blast furnace operation [3]. The huge structure of blast furnace, despite decades of analyses and studies, often behaved like a black box, where various complicated processes that are established during operation defied proper explanation. Until 1950 very little was known regarding the internal structure of the furnace in its running condition. A comprehensive and realistic picture emerged when Japanese researchers froze a running blast furnace by flowing liquid nitrogen through the tuyeres and subsequently dissected the total structure. They thoroughly examined the physical and chemical behaviors inside the furnace and reported their findings to the world [4]. In early stage, mathematical and analytical models were developed to explain the blast furnace phenomena. The zero-dimensional mathematical models were developed initially by using the process as one primary entity [5]. These were followed by models that partitioned the furnace into number of divisions to which different chemical reactions and transport phenomenon were attributed. The Thermodynamic relations were determined by using thermal and chemical conditions [6]. With a view to understand the solid gas temperature variation and chemical composition along the vertical shaft of the furnace, initially one-dimensional models were initiated [7]. Next, these models were augmented by dividing the furnace into finite number of partitions. The rates of chemical reactions were combined with the heat and mass transfer equations and the resulting expressions were utilized to compute the composition and temperature in various sections. This concept was subsequently extended to two and three-dimensional models using computational fluid dynamics and related strategies [8–13]. These models faced difficulty during interpretations of results and the evaluated values are often not close to the interface value, as the actual transport phenomenon and process chemistry are highly complicated inside the furnace. Without directly considering the physics of the process and without directly involving the thermodynamic and transport equations, the new data-driven modeling techniques have evolved in recent years. Through such approaches, the researchers can utilize the gamut of information that the modern blast furnaces routinely capture during their continuous operations and mostly such strategies are intelligent and efficient enough to achieve the necessary goals through intricate computing. In the early stage of data-driven modeling, computing techniques including artificial neural networks and fuzzy logic modeling were usually applied towards the description of the blast furnace processes. These models proved to be a convenient way to calculate the descent solutions for various blast furnace problems and provided descriptive alternatives for meeting the desired requirements and quite importantly, could address the non-linear problems [14–18]. A thorough concept-based accurate model is always crucial for understanding the existing processes and operations in a blast furnace and these modeling techniques significantly contributed towards that. In recent times the computational hardware became tremendously advanced rendering high-performance computing a powerful tool in this domain. Furthermore, the advent of the state of the art techniques such as evolutionary algorithms established a new platform to solve complex industrial problems, including those related to blast furnace. Due to their simple and versatile

nature, these algorithms have some distinct advantages over other optimization techniques. In recent times evolutionary approaches are significantly used in this domain to solve many-objective problems.

The many-objective optimization problems (MaOPs) are special case problems in the domain of optimization where more than three objectives are tackled efficiently. Such problems are ubiquitous in science, engineering and industry and can be formulated in such a way that the solutions are obtained after evaluation of multiple conflicting criteria [19–21]. Many-objective optimization problems have been faced multiple challenges as it includes searching Pareto solutions in multidimensional hyperspace, search strategy and visualization of solutions in multiple front add another level of difficulty in this domain [22]. The optimization problem with multiple numbers of objectives and decision variables are defined as large scale many-objective optimization problems. When evolutionary approaches are adopted to solve many-objective problems, these are known as many-objective optimization evolutionary algorithms (MaOEAs) [23–25]. Initially these algorithms have followed decomposition-based approach. Several challenges are raised during solving these problems. The difficulties are addressed as scalable representation of solutions and visualization algorithm design, and a matrix to evaluate the result. To make them efficient, guided search methods and parameter setting approaches are implemented in this domain [26]. MaOEAs mainly fall under major categories like relaxed dominance-based, indicator-based, aggregation-based, preference-based, reference-based and dimensionally reduction-based approaches. Further details are provided elsewhere [22].

1.1 Challenges in Multi-objective Optimization by Evolutionary Techniques

The researchers have experimented with a wide range of multi-objective and many-objective optimization algorithms to solve nonlinear problems. Initially, they have modified the existing algorithms to increase the capability such that it can handle three or four objectives simultaneously. Li et al. [27] identified the difficulties which are raised during tackling many-objective optimization problems. The situation becomes tough when the number of objectives gets increased, where comparison of non-dominated solutions leads to a challenging situation. The visualization also gets difficult when more than three objectives are associated with a problem. In Pareto based approach, selection pressure arises due to multiple objectives. In aggregation-based approach, weight vector plays a crucial factor. In indicator-based approach the performance indicator, generational distance (GA) and inverted generational distance (IGD) are used in the hypervolume to compute the intensive performance metrics [28]. This leads to high computational cost which renders indicator based approaches difficult to normalize numerous objectives. In reference set approach the difficult task is related to the construction of reference set and creates a balance between

convergence and diversity in reference set. Selection of preference model needs frequent involvement of a decision-maker to get preference information, which is a key issue in preference-based model. These are the recent approaches that are used in many-objective optimization problems. Challenges are eradicated by adding new ideas and modification regularly to the existing algorithms according to the design, condition and structure of the problem.

This chapter is organized as follows. It provides an overview of blast furnace in its initial part. Next, various evolutionary algorithms, which are applied in the blast furnace iron making process are discussed. Thereafter, a brief description of an evolutionary many-objective model, which is currently applied to this field, is included. The prime objective of this application is to find out the multiple numbers of solutions in the multi-objective space. The primary focus of this chapter is to elucidate data-driven models applied in this practical field and to highlight how that helps to solve difficult problems in real time. Finally, conclusions are drawn upon the results of this study and the solving capability of the evolutionary algorithms in the present scenario is evaluated, and a few prospective future issues are highlighted at the end.

2   Blast Furnace

A blast furnace is a huge columnar structure, where iron oxide ore is utilized as raw material added with flux and coke, then melted and produced pig iron as output. The heat exchange and chemical reactions inside the furnace play an important role to produce the hot metal. To generate proper melting temperature air, often enriched with oxygen, is injected through a number of tuyers located at the lower portion of the furnace. The necessary energy and the reducing CO gas are primarily generated through combustion and gasification of coke that is present in the furnace. In modern blast furnaces [4] chute with different angles are used to dump the raw material and flux from the top known as throat of the furnace. Below the throat the raw materials are stacked layer by layer up to a few meters depth, where diameter of this area gradually increases and that portion is called shaft. Just after shaft the diameter is kept constant for a certain depth termed as the belly region. In this region, chemical reaction and heat exchange are carried out and iron ore is transferred into melted liquid iron. From belly to downwards of the furnace is known as bosh region. The hot blast (nitrogen, oxygen) and injectants (pulverized coal, oil) are introduced from this region, which is surrounded by a specific number of tuyeres. A tuyere is a cooled copper nozzle used to supply the required amount of heat and reaction reagents needed for the melting process. The number of tuyeres varies from 12 to 42, according to the size of the furnace [2]. At the lowermost region of the furnace, the hearth is present, where iron and slag are accumulated separately in molten condition. From top to bottom of the furnace the total volume is divided into various zones. The upper portion of the furnace is lumpy zone, where input materials enter and stacking occurs one layer after another. The reduction process is carried out in a cohesive zone, where iron

Fig. 1 Blast Furnace [14]
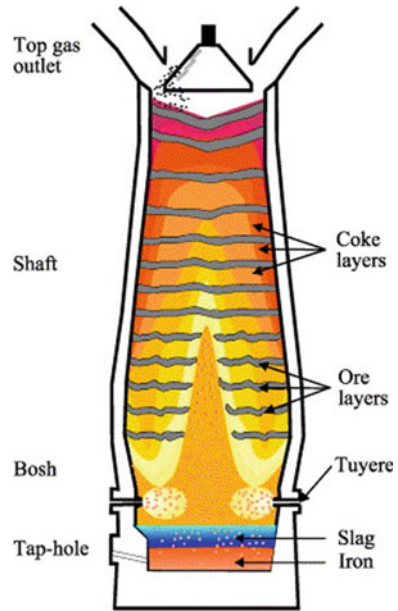
ore (Fe_2O_3) is converted into wüstite (Fe_xO) by reacting with the ascending carbon monoxide (CO) and converted into carbon dioxide (CO_2) after the complete reaction. The iron ore softens and gets melted here. The raceway starts just below the cohesive zone, where coke gasification takes place. In other words, coke is converted into CO by reacting with CO_2 in the presence of inert nitrogen. In the bosh region, unreduced coke accumulates forming the so-called dead man zone. The last zone is named as the hearth. A typical blast furnace is schematically presented in Fig. 1.

2.1 Blast Furnace Modeling and Optimization

The major idea of data-driven blast furnace modeling is to continuously analyze all process data, in order to execute and suggest proper action by taking the guidance of decision-makers in real time. It helps to increase the efficiency of the system and aids to achieve its various targets. As we know that blast furnace is a complex structure therefore, one needs to control a large number of parameters to ensure its smooth running. The objectives like cost, production rate, quality and process efficiency depend upon multiple numbers of variables, which need to be optimized simultaneously. In recent times many-objective optimization [29] has been applied to achieve that. Many-objective optimization modeling helps to find out a number of optimal solutions belonging to a Pareto optimal set [30] which very significantly helps to improve the decision-making process. In such optimization processes, the evolutionary data-driven models have played a major role.

3   Evolutionary Data-Driven Models

In optimization work related to blast furnace data-driven models, Predator-Prey Genetic Algorithm (PPGA), Evolutionary Neural Network (EvoNN), Bi-objective Genetic Programming (BioGP), Evolutionary Deep Neural Network (EvoDN2) and constraint-based Reference Vector Evolutionary Algorithm (cRVEA) have been successfully used for a number of integrated steel plant-related problems [31–38]. These algorithms are efficient enough to compute multiple solutions, which are utilized in the operation process. Further details are provided below.

3.1   Predator-Prey Genetic Algorithm (PPGA)

PPGA [14] is one of the most efficient algorithms that have been used in blast furnace optimization work. The predator-prey algorithm emulates the killing and survival of the animal inhabitants in a forest. A computational grid is generated on which the candidate solutions in the form of preys are placed randomly. In the same way, predators are also placed randomly in the lattice and their designated task is to annihilate the weaker prey. Each predator is associated with a particular weighted sum of the objective functions. There are certain limitations assigned to the movement of the predators, e.g. they can move only for a specific number of steps around a neighborhood, which is also specifically defined. The predator kills the prey present in the neighborhood having the worst fitness. Once the prey is killed it is removed from the grid. If a single prey is present in the predator's neighborhood, then it is killed by default. The predator will take a move in random direction if its neighborhood is empty. The above procedure continues until the completion of pre-specified number of generations. Genetic operators like crossover and mutation are used to produce new individual in each generation. The simultaneous presence of predators favoring each objective allows trade-off solutions to co-exist in the computing space. Instead of associating one objective to each predator, we associate them with different weight vector to maintain diversity in the solution and to obtain good solutions in the Pareto front. A predator-prey two-dimensional grid is shown in Fig. 2.

3.2   Evolutionary Neural Network (EvoNN)

In EvoNN a population of neural nets is used in the evolution process to find out the optimal trade-off between accuracy of the training and the complexity of the networks. A corrected Akaike information criterion (AICc) [14] is used to find out the best-suited model from the solutions belonging to an approximation of the Pareto frontier computed by this algorithm [33]. Here the complexity is determined by the total number of weights associated with the network system, connecting the inputs to
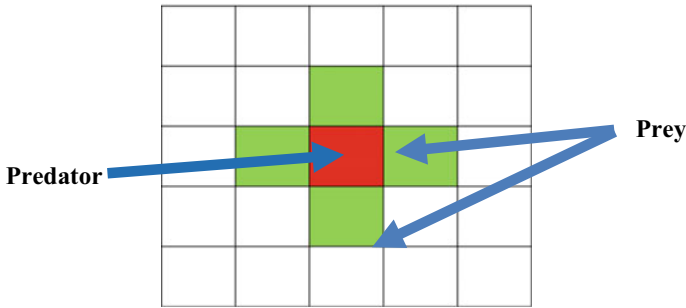
Fig. 2 Predator-Prey in two-dimensional grid

the hidden nodes. In EvoNN the working structure is comprised of two levels, i.e. the lower level and the upper level. Evolution takes place at the lower part, where PPGA is used to optimize the neural nets. In this process, crossover is defined as swapping connections between two neural nets, whereas mutation is performed by perturbing the weights. The amount of change depends upon the values of two randomly selected weights of similar connections. The upper portion of the network, i.e. the output layer uses a linear transfer function and is optimized using linear least squire approach (LLSQ) [14, 15]. This ensures mathematical convergence at the output stage. Further details are provided elsewhere [17].

3.3 Bi-Objective Genetic Programming (BioGP)

In BioGP [16, 17] population of binary trees is used instead of neural network in the evolution process to train the data in order to find out the optimum results. A genetic programming (GP) strategy [39, 40] permits to construct any function based upon the database in hand, by using mathematical operators like addition, subtraction, division, multiplication, etc., as node values in a tree, along with the variables and constants as terminal set values. Unlike conventional genetic programming, here a Pareto tradeoff [16] takes place between the accuracy of training and complexity of the GP trees. The complexity of the GP tree is decided from the level of depth and the spreading of the roots constructed by the user. As indicated before, in the binary trees used here, the operators of the function set are placed at the nodes, while the variables and the constants are placed in the terminal set. The system input can be modeled by evolving mathematical function and can also apply logical condition when it required. The trees are selected based on minimum root mean squire error (RSME) [17]. However, the tree with minimum error may lead to overfitting, on the other hand large error may underfit the data and skip the important trend value in the data set. Here in BioGP the PPGA algorithm is used in the optimization work and a number of smaller trees are aggregated through an LLSQ-based procedure [16, 17]. A typical GP tree is shown in Fig. 3.
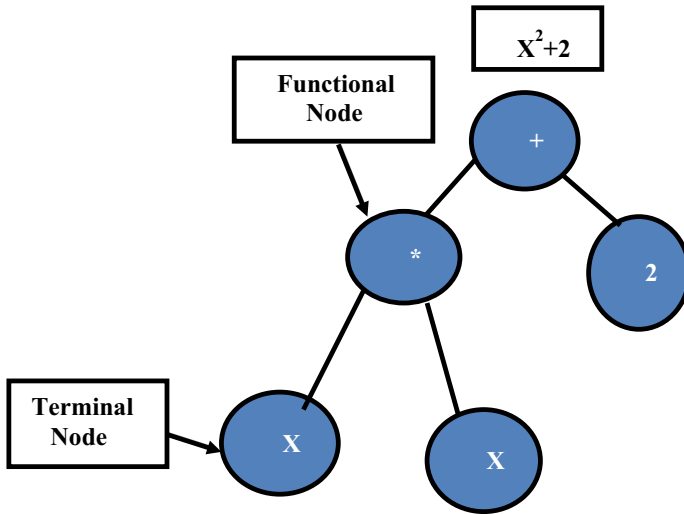
Fig. 3 A typical GP Tree

3.4 Evolutionary Deep Neural Network (EvoDN2)

In data-driven blast furnace modeling often the major challenge is to handle a very large volume of data for which both EvoNN and BioGP are inadequate. EvoDN2 algorithm [41, 42] has been developed to remedy this big data handling problem. This is an intelligent modeling strategy that can accommodate larger data sets with multiple numbers of input variables. In recent times due to its efficient and augmented data handling capability, this technique has been applied to solve the typical problems in blast furnace. In Evolutionary Deep Neural Network architecture a subnet is formed using multiple numbers of hidden layers, where each layer consists of different numbers of nodes. A number of such subnets are aggregated and similar to EvoNN and BioGP, the final convergence is obtained through an LLSQ-based algorithm. It can efficiently manipulate the population and capture many intricate features by a smart modeling strategy, again involving the tradeoffs between the accuracy and complexity of the networks. The inputs from the data set are fed into the subnets randomly, ensuring that each input variable is used at least once. Each subnet, as indicated before, consists of more than one hidden layer as per user's choice. More the number of layers, higher is the accuracy, but it comes at the expense of higher complexity. This can be circumvented by increasing the number of subnets, while individually keeping them small. The crossover and mutation techniques used in EvoNN are inefficient in case of EvoDN2, a number of layers with variable number of hidden nodes are present in the network architecture. Due to larger network size and bigger data set, application of the EvoNN type crossover and mutation would make EvoDN2 computationally very complex and practically render the large optimization task infeasible. This led to development of newer crossover and mutation
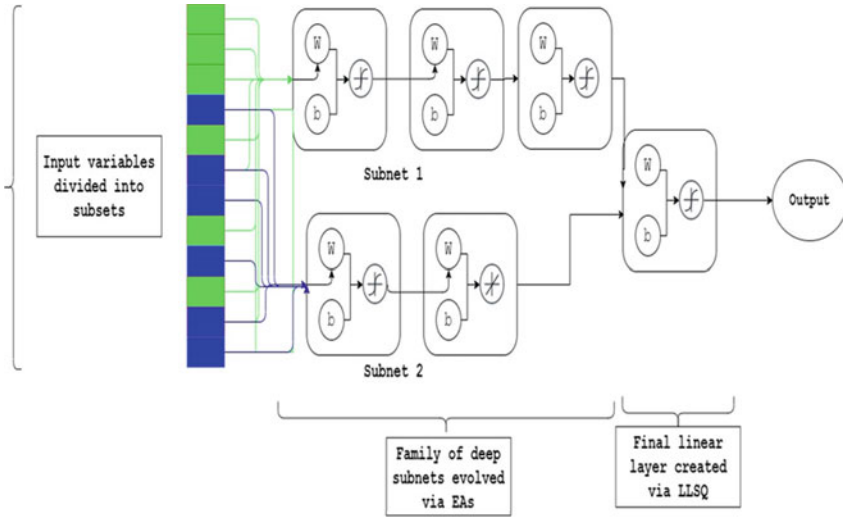
Fig. 4 Schematics of a typical Evolutionary Deep Neural Network

strategies detailed elsewhere [41]. These modifications drastically eliminate the time complexity of the process and similar outcomes are achieved by using simpler mathematical processes compared to the previous techniques [14]. During the crossover between two individuals, a random number of connections are swapped by generating a random number, and the swap is done in parallel, at once, without going over each connection and checking the probability of crossover randomly. This is similar to the older crossover process of EvoNN but acts more efficiently. A slightly altered version of self-adapting mutation technique is used in the mutation process. The outcome results from all the subnets are accumulated and mapped against the output parameters using linear least square method. A schematic deep neural network figure is presented in Fig. 4.

3.5 Reference Vector Evolutionary Algorithm (RVEA)

RVEA algorithm is developed recently to solve multiple numbers of objectives, i.e. more than three objectives in the optimization process [43]. In PPGA a dominance-based procedure is used in the selection process whereas in reference vector evolutionary algorithm a collection of adaptive reference vectors is utilized to reach the Pareto front. An angle penalized distance (APD) is used for the selection process in this algorithm, allowing their reference vectors to converge towards the Pareto front. Here the search process is guided by using a set of reference vectors in the objective space. The initialization of reference vectors is done in accordance to a canonical lattice design and distributed uniformly in the objective hyperspace [36, 44] and after

a prescribed number of generations, expectedly, it converges to an optimal front. At the beginning of each generation the angular distance from each individual to the reference vectors is measured and correspondingly they are assigned to the closest reference vector. The assignment of individual to reference vector is shown in Fig. 5. Then APD is used to select an individual for each reference vector. The convergence and diversity in much objective optimization process are adopted in APD, which is dynamically, a balanced scalarization approach to handle the multiple objectives for the prescribed number of generations quite efficiently. Two things are important in the APD approach. One is the convergence criterion where the distance between the candidate solutions and the ideal point is measured and the other is known as diversity criterion where the significance of acute angle is taken into account, which is measured between the in-focus solution and the reference vector.

The crossover, mutation and adaption of reference vectors continue in each generation until the termination process is completed. A vector to the new optimal front is shown in Fig. 6. For normalization of reference vectors, an adaptive strategy is used to configure the reference vectors in such a way that they can deal with unscalable objective functions. This strategy reconstructs the reference vectors according to the objective functions' ranges and ensures a uniform distribution of the candidate solutions in the objective space.
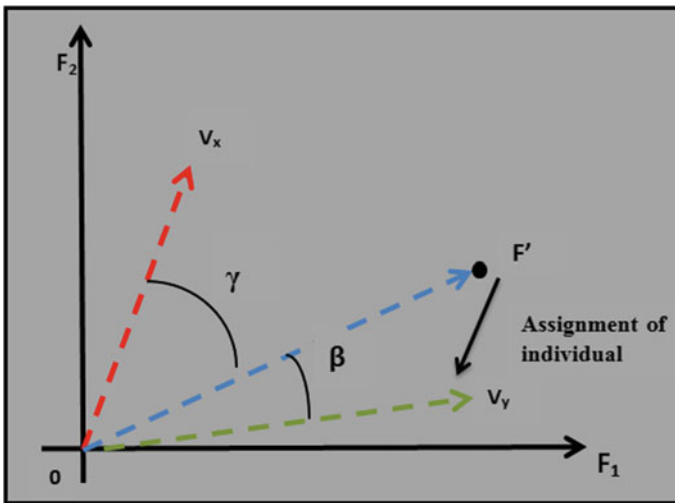
Fig. 5 Assignment of individual to reference vectors
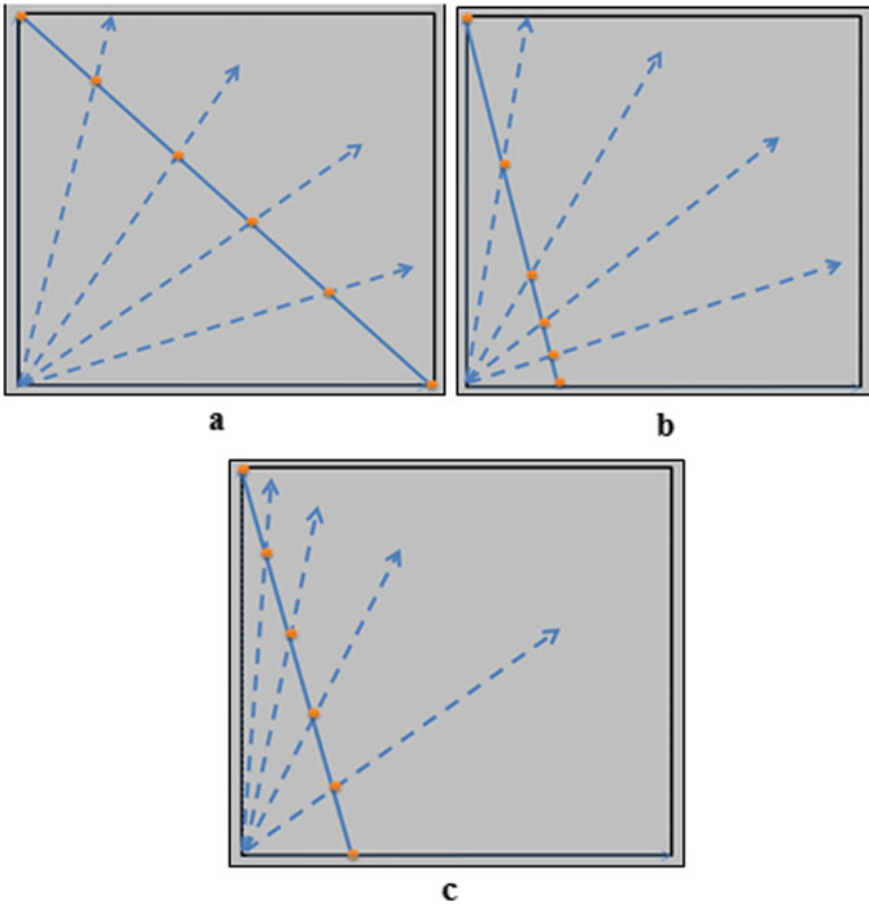
Fig. 6 **a** Uniformly spaced reference vectors with a symmetrical Pareto front result in uniformly dense Pareto solutions, **b** Uniformly spaced reference vector with an unsymmetrical distribution of Pareto solutions which is undesirable, **c** Equal distribution is achieved by reference vector adaption

4 Handling a Real-World Blast Furnace Iron Making Problem Using Evolutionary Approaches

4.1 Objective and Scope

In this section, applications of evolutionary deep learning algorithm EvoDN2 to generate the training models in a blast furnace scenario will be described, along with the models constructed from other data-driven evolutionary approaches like BioGP and EvoNN. Blast furnace processes will be represented as a time series problem, where the current value of the parameters involved have certain contributions from

the same parameters used in the past. Such situations would be considered extensively and time-lagged variables would be incorporated to the parameters. Using this, it will be demonstrated how one can produce much better results in terms of modeling and optimization of the complex time-dependent blast furnace processes. The primary focus has been to use the novel EvoDN2 to mimic that reactor and study how the results fare against the existing algorithms, EvoNN and BioGP in particular. The use cRVEA for the optimization of objectives obtained through the trained models will be demonstrated as well, followed by a detailed analysis of the findings.

4.2 Data Preparation

In this research work, operational data consists of daywise input and output information, including both measured and computed values, of an operational blast furnace. By studying the individual process parameters, a data-driven strategy is developed. The influencing parameters which directly or indirectly affect the system are considered as decision variables and the functions important from the process point of view, which depend open these variables are taken as the objectives that required to be optimized. The operational dataset used here consists of eight input variables and four output objectives. The industrial data sheet provided five consecutive years of operational information. These data are efficiently handled by removing the outliers and adding the missing input and output values by using a K-Nearest Neighbors (KNN) approach [45, 46]. The data set contains variables describing information regarding input materials' physical and chemical properties, control parameters, reducing agents, and internal state of the blast furnace. By considering these variables and objectives, data-driven models are derived for subsequent optimization. The operational decision variables in the blast furnace are taken as (i) basicity, (ii) hot blast volume (iii) hot blast pressure, (iv) hot blast temperature, (v) iron ore in the charge, (vi) sinter in the charge, (vii) coke rate and (viii) silicon in the hot metal. The amount of input variables distribution depends upon the process and the state of the furnace operation. The influenced objectives are taken as (i) CO/CO_2 ratio (ii) oxygen enrichment, (iii) productivity and (iv) raceway adiabatic flame temperature (RAFT). In Table 1 the ranges of the available raw data are presented for a five years operational period of the steel plant.

4.3 Time Series Modeling

Time series modeling [47] works as an effective procedure for dealing with past data analysis in order to predict future result. For this strategy to be effective, past data should be carefully handled and a proper study is needed to develop an exact model representing the inherent structure of the collected data. The model is then used to generate a future value for the series to be forecast. This is essentially predicting

Table 1 Range of data used in data-driven model

Input parameters	Data range				
	Year 1	Year 2	Year 3	Year 4	Year 5
Basicity (X1), ratio	0.87–1.07	0.89–1.06	0.56–1.11	0.61–1.27	0.7–1.23
Hot Blast Volume (X2), Nm^3	1097–2686	832–2660	900–2660	1530–2660	710–2630
Hot Blast Pressure (X3), Kg/cm^2	1.14–2.72	0.73–2.68	1.65–2.78	1.9–2.84	1.15–2.89
Hot Blast Temperature (X4), °C	914–1203	963–1209	850–1200	1000–1214	830–1200
Iron Ore (X5), Mt	0.24–0.811	0.36–0.813	0.433–.932	0.365–738	0.192–1.348
Sinter (X6), Mt	0.34–1.52	0.807–2.04	0.873–2.04	0.849–1.245	0.77–2.54
Coke (X7), Kg/thm	360–745	337–1105	313–983	307–606	273–884
Si (X8), %	0.25–1.98	0.02–1.98	0.28–2.73	0.29–1.27	0.15–3.22
CO/CO_2 (Y1), ratio	1.12–1.70	1.12–1.64	1.02–1.63	1.09–1.30	1.08–3.87
O_2 Enrichment (Y2), Nm^3/thm	0.04–68.85	0.009–69.25	0.002–4.80	0.08–5.61	0.009–5.15
Productivity (Y3), Mt/d/m3	0.14–2.49	0.75–2.41	0.136–2.83	1.16–2.90	0.064–2.93
RAFT (Y4), 0C	2010–2330	1950–2575	1860–2510	2000–2410	2040–2566

the future trend by understanding the past. This technique has been used largely in diverse areas such as banking, economic activity, as well as in industry, science and engineering. Recently artificial neural networks (ANN) and genetic programming (GP) [48] have received significant attention in time series forecasting. These techniques are capable of handling nonlinear data without any prerequisites on the statistical distribution that the observation follows: these techniques are data-driven and at the same time self-adaptive in nature. In the context of blast furnace iron making, such procedures are added to EvoNN, EvoDN2 and BioGP in the training process to reduce the error and increase the correlation coefficient.

4.4 Many Objectives Optimization Formulation

The objective functions are shown in Table 2. The goal of this work is to minimize CO/CO_2 ratio. The emission of CO and CO_2 is basically due to burning of coke and fuel inside the blast furnace [49, 50]. Coke acts as a reducing agent it also helps to increase the adiabatic flame temperature but improper burning of coke increase the amount of unreacted CO at the top gas leading to inefficient usage of the reducing gas and fuel. Similarly, it may lead to an increased amount of CO_2, which is undesirable from the environmental point of view. Oxygen enrichment is needed to reduce the emission rate of this pollutant gas, at the same time it increases the adiabatic flame

Table 2 Objective function formulation

Objectives	Task
CO/CO$_2$ (Y1)	Minimize
O2 Enrichment (Y2)	Minimize
Productivity (Y3)	Maximize
RAFT (Y4)	Maximize

temperature and thus has a positive impact on productivity. The oxygen enrichment is usually done in three levels. Less than 28% is considered the lower level, between 28% and 45% is the medium level and more than 45% is taken as the higher level [51]. Oxygen enrichment technology directly affects the cost of the process. Expectedly, higher oxygen enrichment in a process involves more cost. On the other hand, to generate higher amount of heat higher quantity of oxygen is needed. The raceway adiabatic flame temperature (RAFT) controls the rate of chemical reaction in the combustion and reduction process in the blast furnace [52]. However in operational condition, to maintain the optimum flame temperature often is a major challenge. The dropping of flame temperature directly affects melting capacity and reduction process. Due to this effect the thermal heat balance of the process also reduces, which makes the system unstable. It means RAFT must be maximized to achieve the optimal condition. Three other objectives were considered in this study as shown in Table 2, out of which one required maximization and the remaining two were candidates for minimization. These four objectives are interrelated to each other and during their simultaneous optimization; any improvement in one objective directly impacts the remaining ones. Handling such type of conflicting objectives in a process leads to a difficult task of computing their Pareto optimality. Here we tried to formulate a multi-objective optimization scheme according to the requirements of blast furnace iron making process.

4.5 Construction of Metamodels

In this study, construction of data-driven metamodels involves a bi-objective optimization task of their accuracy and complexity and this is done by considering various levels of constraints. The training is carried out using the nonlinear and noisy data, separately for each objective, for a prescribed number of generations. Two types of training models are generated. One is without considering the time lag another with consideration of time lag. The training models are generated separately for CO/CO$_2$ ratio, Oxygen enrichment, RAFT and Productivity. In EvoNN and EvoDN2 the corrected Akaike information criteria is used to find out the best model out of those generated in the training process but in case of BioGP, the model with least training error is automatically picked up after the training process. The accuracy versus model complexities is found out from the output results of the training models.
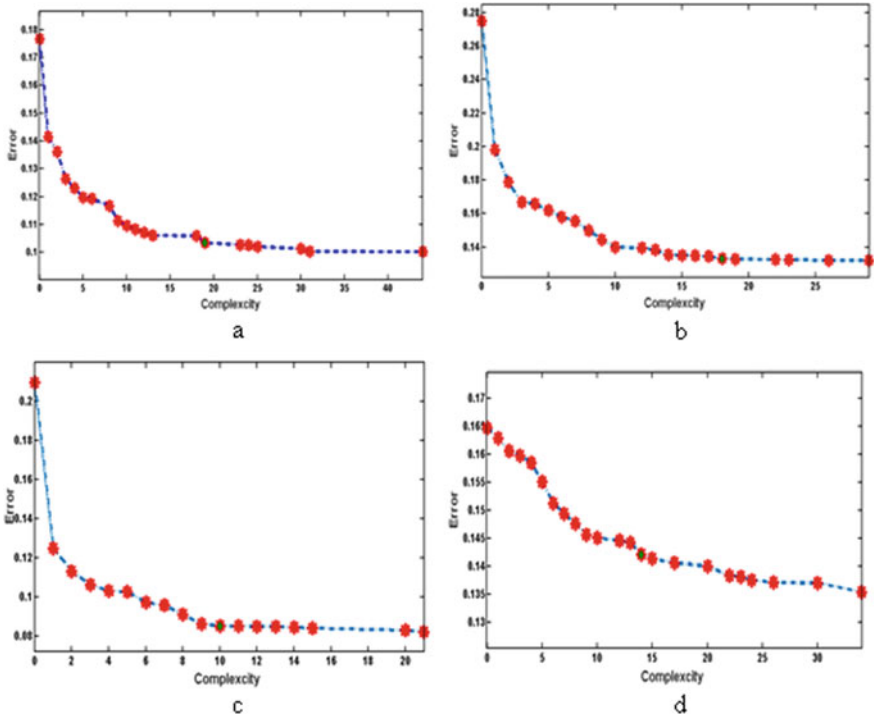
Fig. 7 a–d Training models for CO/CO$_2$, Oxygen Enrichment, Productivity and RAFT by using EvoNN

A typical result is shown in Fig. 7 where each point denotes a separate model with its unique topology and architecture.

During the training work, the industrial data are run up to prescribed number of generations. After completion of training, a Pareto tradeoff is generated between accuracy and complexity of the models. In EvoNN and EvoDN2, the model from the Pareto frontier is selected on the basis of lowest AICc value and in case of BioGP the tree producing the least training error is selected. The parameters of the data-driven models which are set for the training of the individual are shown in Tables 3 and 4.

4.6 Training and the Correlation Coefficients

In evolutionary many-objective optimizations various algorithms like EvoNN, BioGP and EvoDN2 are used to generate training models from the real data of blast furnace. Here the physical information about the process is generated only through the plant data without using any theoretical models. These data which are accumulated during the routine operation of the furnace may not have always belonged to the specific

Table 3 Parameters of the data-driven models using EvoNN and BioGP

Parameters	Algorithm EvoNN	Algorithm BioGP
Number of generations	150	100
Maximum rank	20	20
Maximum number of roots	NA	5
Maximum depth	NA	5
Killing interval	7	7
Number of preys	500	500
Number of predators	50	50
Grid size	60*60	60*60
Hidden nodes	5	NA

Table 4 Parameters of the data-driven model using EvoDN2

Parameters	Algorithm EvoDN2
Number of generations	100
Number of subnets	3
Number of hidden nodes	9,8,8
Maximum rank	20
Killing interval	5
Number of preys	100
Number of predators	40
Grid size	20*20

desirable ranges, sometimes fluctuations above or below such limits happen due to various factors including reaction complexity and the material behavior found inside the furnace. To predict an exact model out of these nonlinear and noisy data intelligent training through the above-mentioned algorithms is necessary. These strategies are configured in such a way that they can develop a novel model, which neither over fits nor under fits with the actual data; an actual representative model is thus most likely to emerge. After the learning gets over, the correspondence between the industrial operational data and the model-generated data can always be evaluated through the correlation coefficient.

The previously discussed parameters are used in the training process to generate the training model. After training work, the output results are evaluated. Some typical training and correlation coefficient figures generated through the algorithms are shown in Figs. 8, 9 and 10.

By analyzing these training and correlation curves, it becomes clear that the data generated from the trained models neither underfit nor overfit the industrial data. In most cases, the correlation coefficient resulted from the individual training of the objectives turned out to be more than 60%. Out of the three strategies EvoDN2,
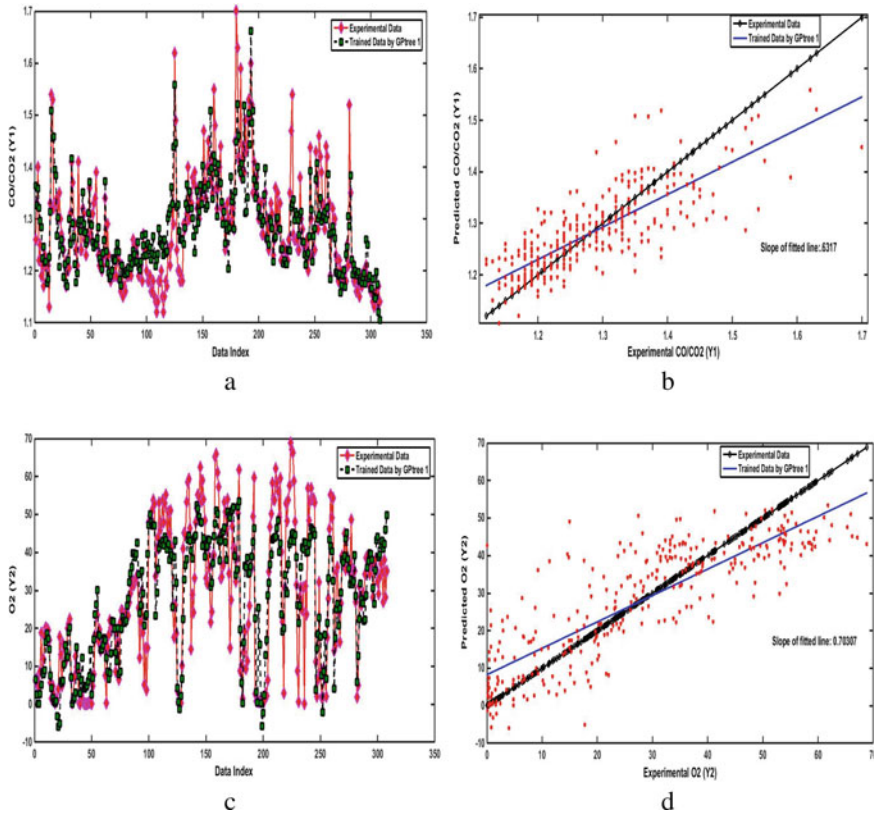
Fig. 8 a–d Training and correlation coefficient figures generated by data-driven model (BioGP)

expectedly, had shown a better correlation even then any very tight fitting was avoided as that might lead to capturing the noise in the data. Here the training models are generated by considering two methods, one by using time lag variables and another without time lag. The training result evaluated from time lag concept is 5 to 8% better than the result generated without time lag in terms of the correlation values. The results are shown in Table 5. In this work therefore the models generated with time lag were used subsequently in the optimization work.

4.7 Single Variable Response (SVR)

The behavior of each decision variable and their influencing characteristics on the objectives can be determined in these models through a simple procedure known as single variable response (SVR) [53, 54]. As these variables significantly affect each other, more often it is critical to isolate the actual role played by an individual
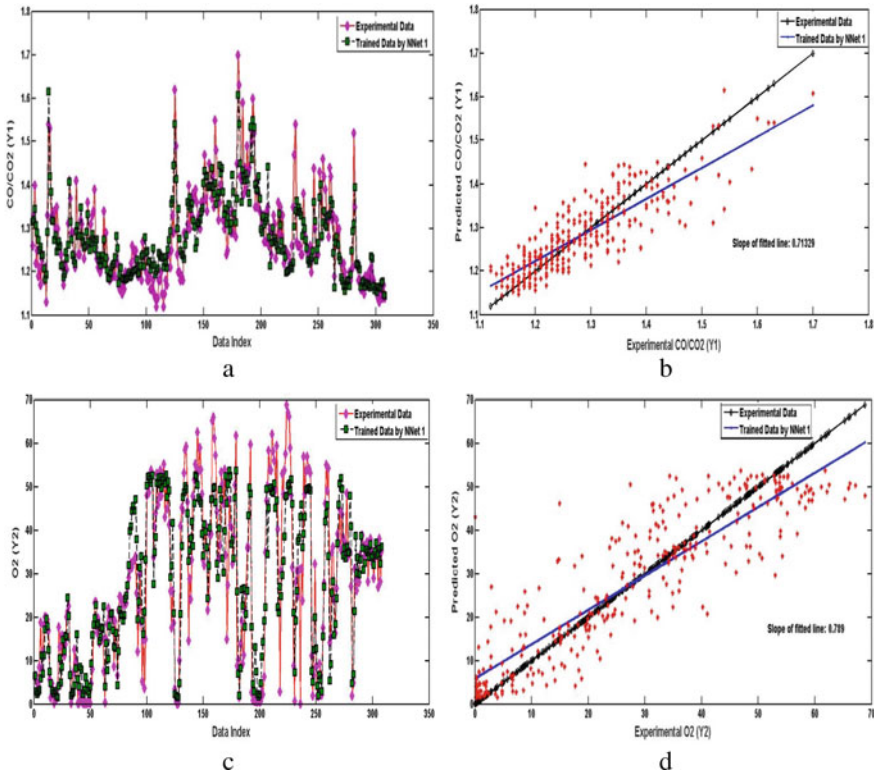
Fig. 9 a–d Training and correlation coefficient figures generated by data-driven model (EvoNN)

parameter. This simple strategy was developed and incorporated in our work to evaluate the dependable characteristic of any individual parameter with the others. It involved providing some arbitrary but systematic perturbation to each variable at a time, in the trained model, keeping the rest of the decision variables constrained at the base level. The corresponding response of the output is recorded and examined. The arbitrary perturbation includes increase and decrease in the value of the variable being examined, which includes sharp changes as well as slow and gradual changes, along with some constant regions. If the objective shows similar response as the input, then it is taken as directly correlated (+ve response) with the output, similarly for a reverse response of exact opposite trend of the output pattern an inverse correlation (−ve response) is assumed. Sometimes no such correlation could be detected and such cases are also systematically noted. In few occasions both +ve and −ve patterns are found, which is described as a mixed response. The individual behavior of each variable with respect to output is thus known as the single variable response.

By using EvoNN, BioGP and EvoDN2 single variable response of all the variables are generated. In each model the considered variables are basicity, blast furnace volume, hot blast pressure, hot blast temperature, the amounts of iron ore, sinter,
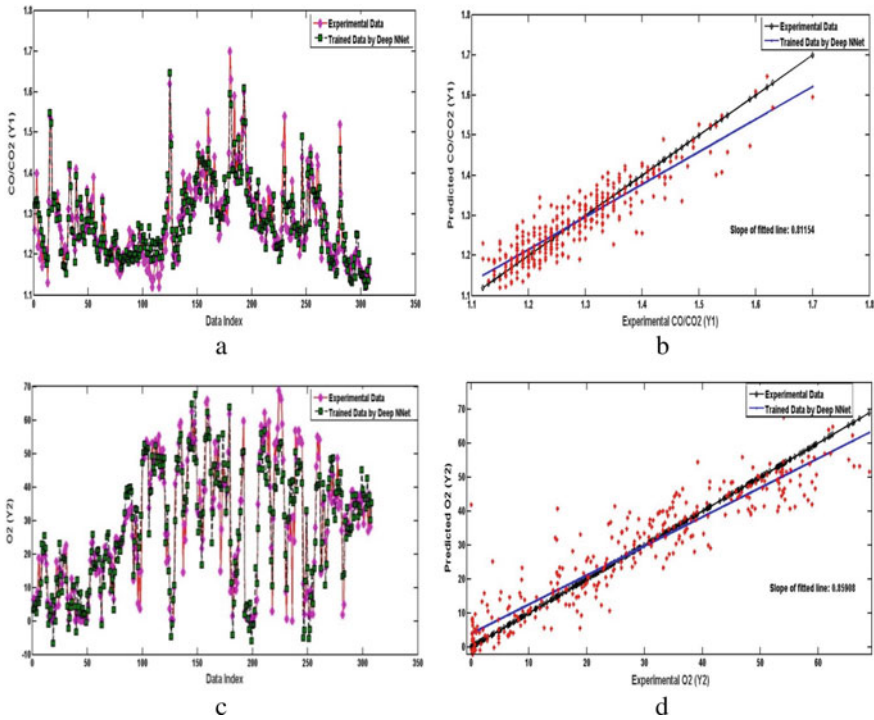
Fig. 10 **a–d** Training and correlation coefficient figures generated by data-driven model (EvoDN2)

coke, SiO_2 and the percentage of silicon in the hot metal, which directly, affect the objectives like CO/CO_2 ratio, oxygen enrichment, productivity and RAFT. The evaluated results from the model signify the influence of the individual variables on a particular objective during the operation of the furnace. The outcome of the results show that the behavioral tendencies of the variables can be classified as +ve, −ve, mixed and no response. Some typical response curves generated through this procedure are shown in Fig. 11, where 1 for direct response, −1 for inverse response, 2 for mixed response, and 0 for no response. From the figures a clear visualization of input signal with the output is obtained.

From EvoNN model the evaluated results show that basicity has no impact on the CO/CO_2 ratio, −ve impact on oxygen enrichment, +ve impact on productivity and mixed response to race way adiabatic flame temperature (RAFT). In BioGP the training model of the evaluated result shows a −ve response in case of CO/CO_2 ratio, no detected impact on oxygen enrichment, +ve impact on productivity and a mixed response to adiabatic flame temperature. In EvoDN2 the same variable shows −ve response to CO/CO_2 ratio, −ve response to oxygen enrichment, mixed impact on productivity and +ve impact on RAFT. In most of the cases, input variable and output shows similar type of responses but in few cases they behave differently from one model to another. This is primarily because SVR is an approximate method and

Table 5 Correlation coefficients with and without time lag for different evolutionary models

Year	Model	Correlation coefficient without time lag				Correlation coefficient with time lag			
		CO/CO$_2$ (Y1)	O$_2$ (Y2)	Productivity (Y3)	RAFT (Y4)	CO/CO$_2$ (Y1')	O$_2$ (Y2')	Productivity (Y3')	RAFT (Y4')
1	BioGP	0.61	0.63	0.92	0.34	0.63	0.70	0.87	0.36
	EvoNN	0.64	0.73	0.93	0.32	0.71	0.78	086	0.36
	EvoDN2	0.74	0.78	0.95	0.33	0.81	0.85	0.92	0.52
2	BioGP	0.49	0.39	0.89	0.40	0.60	0.52	0.92	0.46
	EvoNN	0.50	0.48	0.90	0.36	0.63	0.59	0.94	0.47
	EvoDN2	0.56	0.52	0.92	0.53	0.72	0.70	0.96	0.62
3	BioGP	0.60	0.69	0.89	0.58	0.97	0.97	0.95	0.65
	EvoNN	0.65	0.68	0.89	0.62	0.98	0.97	0.95	0.67
	EvoDN2	0.74	0.75	0.92	0.70	0.98	0.98	0.97	0.79
4	BioGP	0.32	0.73	0.71	0.55	0.39	0.72	0.81	0.61
	EvoNN	0.38	0.75	0.76	0.54	0.59	0.80	0.81	0.59
	EvoDN2	0.53	0.81	0.79	0.65	0.69	0.86	0.89	0.71
5	BioGP	0.99	0.83	0.93	0.56	0.99	0.84	0.93	0.62
	EvoNN	0.99	0.86	0.94	0.58	0.99	0.86	0.94	0.68
	EvoDN2	0.99	0.87	0.95	0.63	0.99	0.92	0.97	0.75
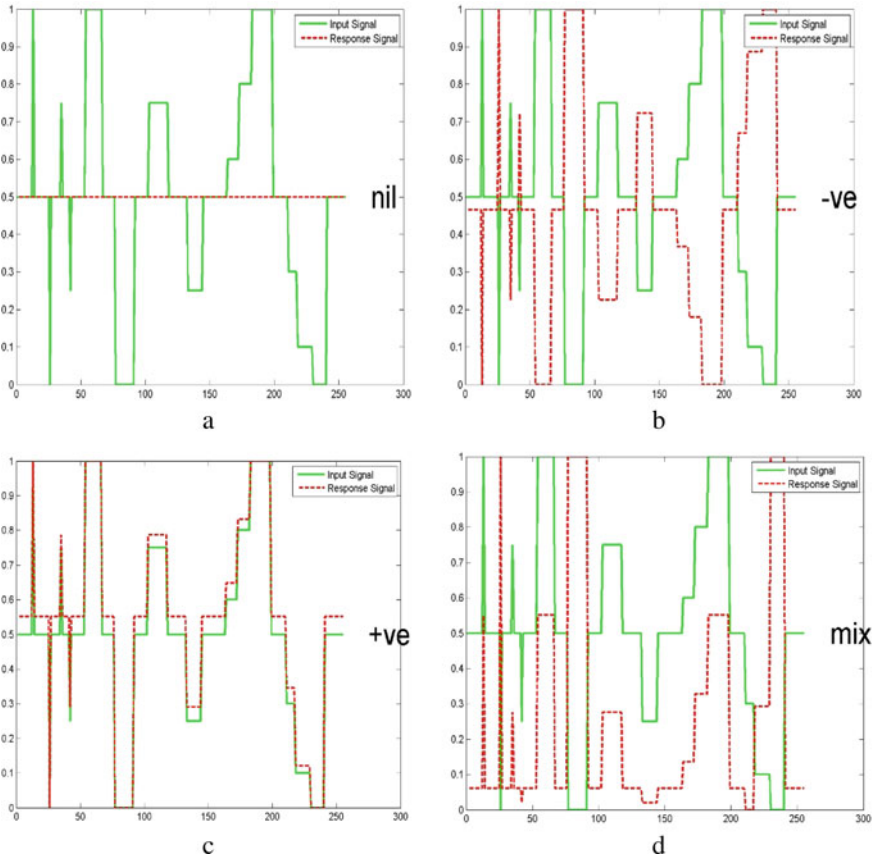
Fig. 11 **a–d** The single variable response figures generated from EvoNN

also often a particular strategy is not capable of capturing every trend of the data in hand. The model generated responses are summarized in Table 6. The results are

Table 6 Single variable responses generated from evolutionary models

Single Variable Response (SVR)													
EvoNN	Y1	Y2	Y3	Y4	BioGP	Y1	Y2	Y3	Y4	EvoDN2	Y1	Y2	Y3	Y4
X11	0	−1	1	2	X11	−1	0	1	2	X11	−1	−1	2	1
X12	0	−1	2	0	X12	−1	0	2	0	X12	2	−1	2	2
X13	0	−1	0	1	X13	−1	0	1	0	X13	−1	2	0	2
X21	0	2	2	0	X21	−1	−1	2	0	X21	−1	−1	1	2
X22	0	2	−1	−1	X22	0	2	2	0	X22	2	2	2	−1
X23	0	0	2	2	X23	1	0	0	2	X23	2	−1	2	2

very convenient to identify the trends during actual operation of the furnace. In this way, all the input variables have been tested against all the objectives and their trends and behavior are examined properly. From this point onward Xij will denote the jth time lagged value of the variable Xi.

4.8 Optimization Work

Optimization plays an important role to generate practically useful solutions from the models, which are evaluated from the training algorithms EvoNN, BioGP and EvoDN2. Here specially constructed neural net, genetic programming and deep neural nets are used to find a suitable optimum model from number of such models present in the Pareto frontier of the respective training algorithm. Next, the selected optimum models are used in the multi-objective optimization task already shown in Table 2. As evident from that table, here four objectives are considered at a time. In this many-objective problem, reference vector evolutionary algorithm is used to compute the result. The individual optimum training models of all the objectives are processed through constraint-based reference vector evolutionary algorithm (cRVEA). As mentioned before, this algorithm uses uniformly distributed reference vectors and the candidate solutions in the problem are assigned to the suitable reference vectors and the population changes in a standard evolutionary way. One individual is selected for each reference vector according to the angle penalized distance (APD). Any individual convergence and divergence is decided from APD, as discussed earlier. In this optimization process four basic steps, namely, the generation of reference vectors, assignment of individual to reference vectors, selection and adaption of reference vectors are applied in the evaluation process to find out the best suited solutions. The models obtained through EvoNN, BioGP and EvoDN2 are processed through this optimization process. Below, the results and the multi-dimensional figures generated through this algorithm are now discussed in detail.

The optimized result generated from the industrial data contains daywise information for consecutive five years. For the optimization purpose data-driven models are constructed year-wise. Individual year of data is denoted as Year1 to Year5. All these data are separately trained and modeled by using EvoNN, BioGP and EvoDN2. In each case, the models were selected from their respective Pareto fronts consisting of a tradeoff between complexity and accuracy of training. The cRVEA algorithm is applied in this many-objective problem, where, as mentioned before, CO/CO_2 ratio, oxygen enrichment, productivity and RAFT are considered as objectives for the problem. These objectives are optimized simultaneously at a time by considering year-wise training data. Here the objectives are designated as Y1, Y2, Y3 and Y4 for quick reference. The trained objectives are utilized in the optimization process, generating results in a multi-dimensional space. Figures 12 and 13 show optimized results for Year1 and Year2 in a many objective space. In Fig. 12, the optimized results generated from cRVEA algorithm by using Year1 trained data from EvoNN,
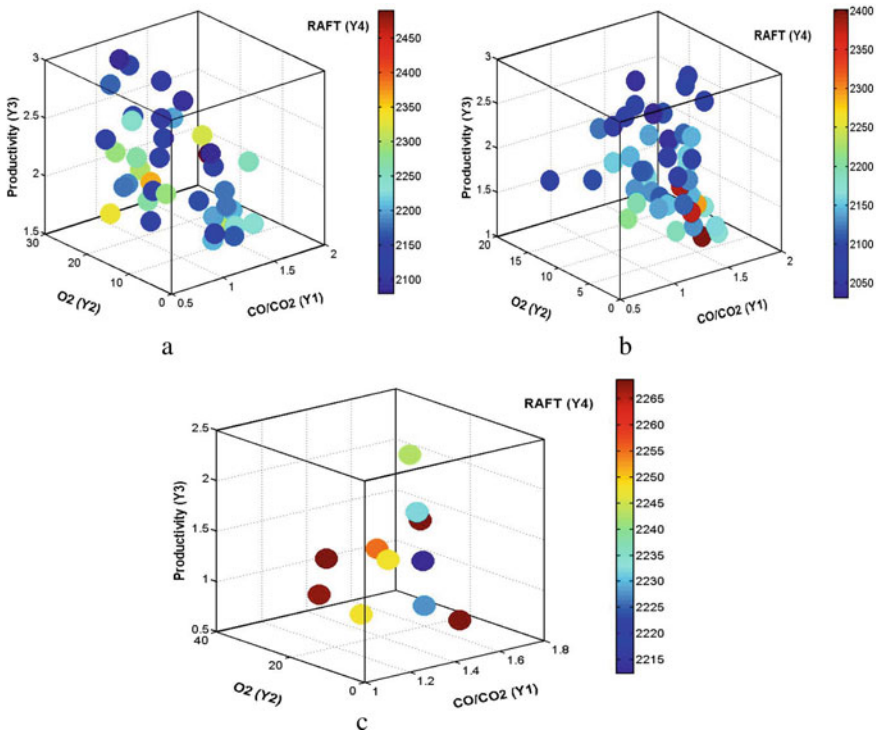
Fig. 12 cRVEA Optimized results generated using **a** EvoNN, **b** BioGP and **c** EvoDN2 on Year 1 training data

BioGP and EvoDN2 are presented. These algorithms have been applied to the models obtained from the other sets of data as well.

Evolutionary algorithms dealing with several objectives require viable methods to usefully visualize and represent a multi-dimensional set of solutions. Parallel coordinates plotting [55], which works well for high dimensional data, has been frequently used in such occasions. To show a set of points in a multi-dimensional space, parallel coordinates map them onto a 2D graph, where multiple parallel axes are plotted, typically in a vertical and equispaced fashion. These vertical axes represent the objectives and scaled properly. Drawing a locus of the objective values of a particular solution represented in this fashion enables its simple but effective visualization. Thus parallel plotting provides a clear idea regarding the obtained result. Here Figs. 14 and 15 show the parallel plots for optimized resulst using the models for Year1 and Year2 data.
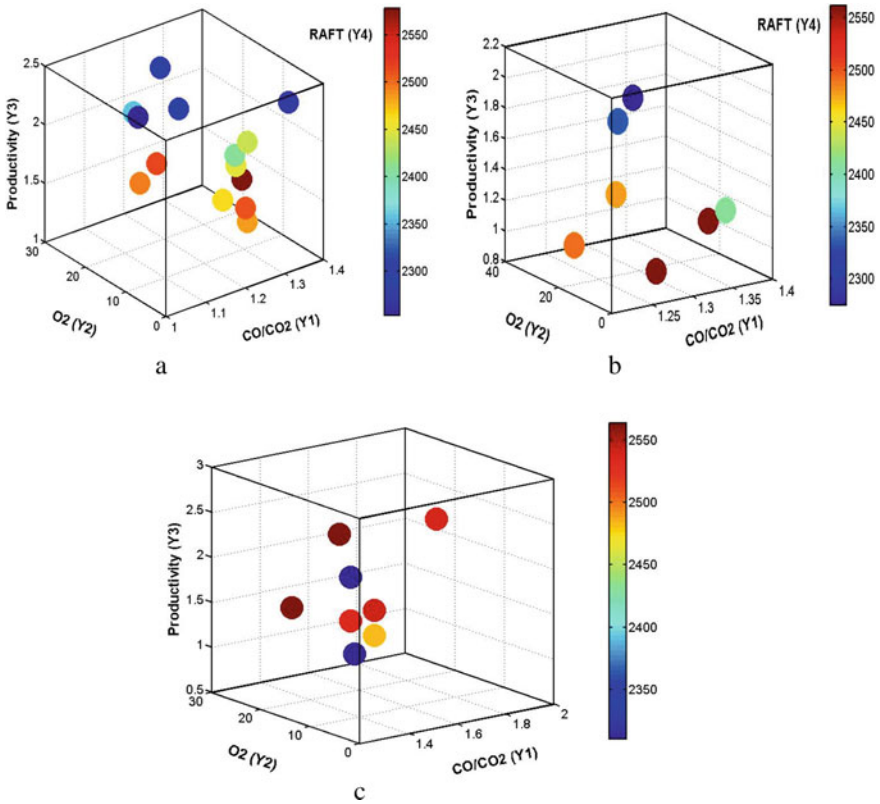
Fig. 13 cRVEA Optimized results generated using **a** EvoNN, **b** BioGP and **c** EvoDN2 on Year 2 training data

4.9 Result Analysis

Results obtained from the optimization process by using cRVEA are presented in Table 7. Since a very large number of Pareto optimum solutions are obtained, only their maximum and minimum values are indicated in the table, in order to compare the corresponding ranges in the original data set. The analysis of the results is based upon three things. Firstly, a comparison of year-wise optimized results obtained from all the data-driven modeling techniques is made with the actual industrial data. Secondly, a number of Pareto solutions generated from optimization of trained models from Year1 to Year5 datasets and their ranges of solutions are evaluated. Thirdly, a comprehensive analysis of the Pareto solutions generated from the total five years' data set is conducted and their impact on the decision-making process is explored.

Let's recall that during the optimization process cRVEA algorithm has been applied to the objectives like CO/CO_2 ratio, oxygen enrichment, productivity and RAFT, where all of them are optimized simultaneously. The original industrial data
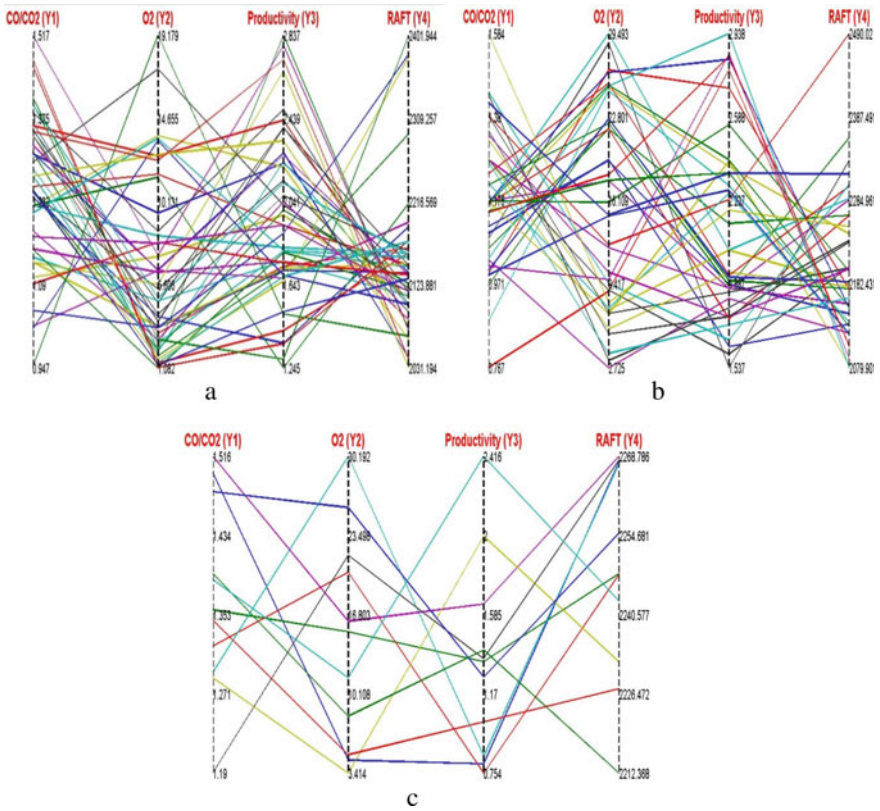
Fig. 14 Parallel plots of optimized results generated from **a** EvoNN, **b** BioGP and **c** EvoDN2 using Year 1 training data

indicate that from Year1 to Year5 all the objective ranges are quite high, which are not conducive of a very efficient performance, and further highlights the need for optimization. From Table 7 it is seen that in the original data the range of CO/CO_2 ratio is obtained between 0.02 and 3.27, oxygen enrichment is found between 0.01 and 69.25 Nm^3/thm, productivity is given between 0.04 to 2.93 mt/d/m^3 and the raceway adiabatic flame temperature is recorded between 1860 and 2575 °C. These are the total ranges of variation of individual objectives which are already tabulated with details. According to the optimization formulation the objective like CO/CO_2 ratio and oxygen enrichment is to be minimized, on the other hand productivity and RAFT are to be maximized. These strategies are followed inside cRVEA algorithm to find out optimum solutions. In initial case, the BioGP trained models of all the objectives are run through the cRVEA algorithm. The evaluated optimized result shows that the range of solutions for CO/CO_2 ratio is generated between 1 and 1.59, oxygen enrichment is in between 0.03 and 29.5 Nm^3/thm, productivity is in between 1.13 and 2.94 mt/d/m^3 and RAFT is in between 2078 and 2554.47 °C. Similarly,
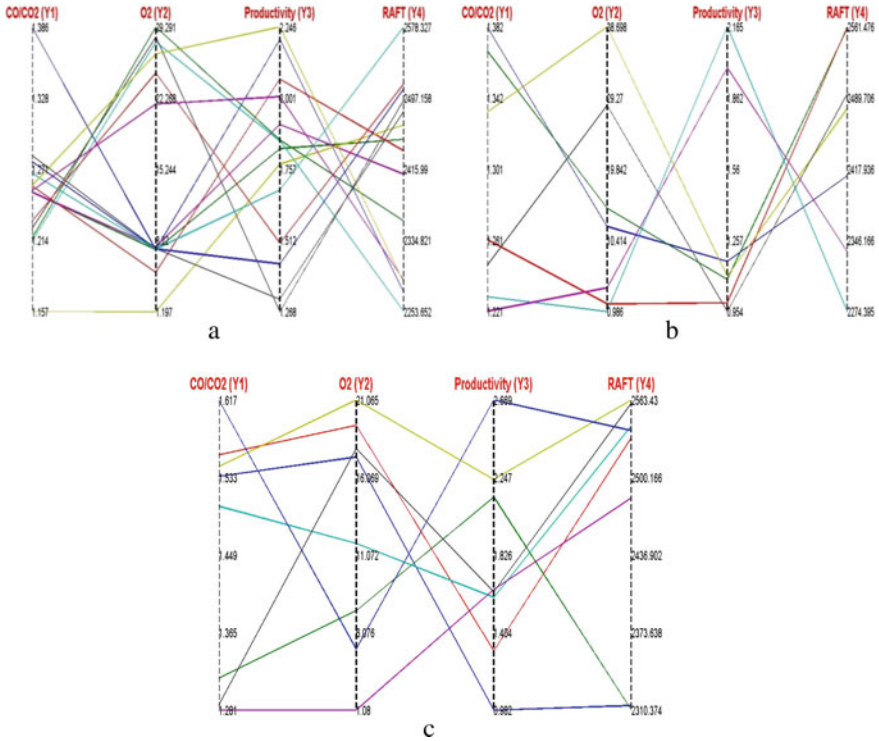
Fig. 15 Parallel plots of optimized results generated from **a** EvoNN, **b** BioGP and **c** EvoDN2 using Year2 training data

the optimum results generated from the EvoNN trained data are also well within the range. The range of solutions for CO/CO_2 ratio is generated between 1 and 1.52, oxygen enrichment is in between 0 and 25 Nm^3/thm, productivity is in between 1.08 and 2.84 mt/d/m^3 and RAFT is in between 2031 and 2583.41 °C. The optimized result for EvoDN2 trained data also occurred within the required limit. The range of solutions for CO/CO_2 ratio is between 1 and 1.61, oxygen enrichment is in between 0 and 22 Nm^3/thm, productivity is in between 1.2 and 2.83 mt/d/m^3 and RAFT is in between 2100.3 and 2596.39 °C. Year1 and Year2 optimized solution ranges are shown in Figs. 16 and 17 where a clear visualization of the spread of optimized results with respect to the original industrial data is readily available.

A close examine the optimal solutions generated from the trained models by using constraint-based reference vector evolutionary algorithm, indicates that the range of solutions and number of Pareto points vary significantly from one algorithm to another. These optimum points satisfy the optimization rational that is defined earlier. Expectedly, in a minimization problem the objective values are drifted toward their lower bound and the reverse happens for a maximization problem, where the optimized objective values are drifted toward maximum range of the industrial data

Table 7 Results evaluated by using cRVEA algorithm

Year	Algorithm	Objectives							
		Y1 (CO/CO_2)		Y2 (O_2)		Y3 (Productivity)		Y4 (RAFT)	
		Min	Max	Min	Max	Min	Max	Min	Max
1	Original Data Sheet	1.12	1.70	.04	68.85	0.14	2.49	2010.00	2330.00
	BioGP_cRVEA	0.99	1.58	2.72	29.49	1.53	2.93	2079.90	2490.02
	EvoNN_cRVEA	1.06	1.65	1.08	19.17	1.24	2.83	2031.00	2401.94
	EvoDN2_cRVEA	1.19	1.51	3.41	21.89	1.3	2.6	2212.36	2268.78
2	Original Data Sheet	1.12	1.64	0.01	69.25	0.75	2.41	1950	2575
	BioGP_cRVEA	1.22	1.38	0.98	28.69	1.15	2.16	2274.39	2561.47
	EvoNN_cRVEA	1.15	1.38	1.19	24.49	1.26	2.24	2253.64	2578.32
	EvoDN2_cRVEA	1.28	1.61	1.08	21.06	1.25	2.66	2310.37	2563.43
3	Original Data Sheet	1.02	1.63	0.002	4.80	0.14	2.83	1860.00	2510.00
	BioGP_cRVEA	1.27	1.59	0.21	3.49	1.13	2.09	2225.47	2508.94
	EvoNN_cRVEA	1.22	1.41	0.06	3.35	1.16	2.08	2292.65	2509.32
	EvoDN2_cRVEA	1	1.35	0.08	2.22	1.73	2.17	2222.39	2596.39
4	Original Data Sheet	1.09	1.30	0.08	5.61	1.16	2.90	2000.00	2410.00
	BioGP_cRVEA	1.05	1.20	0.03	3.98	1.35	2.90	2123.38	2543.51
	EvoNN_cRVEA	1.01	1.14	.001	3.84	1.32	2.82	2104.33	2583.41
	EvoDN2_cRVEA	1.10	1.27	0.21	5.08	1.21	2.82	2100.30	2491.93
5	Original Data Sheet	1.08	3.87	0.01	5.15	0.07	2.93	2040.00	2566.00
	BioGP_cRVEA	1.19	1.59	0.33	5.00	1.16	2.12	2233.57	2553.54
	EvoNN_cRVEA	1.15	1.51	0.02	2.94	1.08	2.31	2100.60	2325.51
	EvoDN2_cRVEA	1.08	1.60	0.12	4.61	1.24	2.36	2339.57	2560.39

used. In this problem, the oxygen enrichment in the industrial data sheet is in the range from 0 and 69 Nm^3/thm, which is to be minimized according to our formulation. Looking at all the models we observe that the optimized solutions are clustured in a narrower range toward minimum side of the objective space: in EvoNN models, the optimized range is between 0 and 25 Nm^3/thm, for BioGP models the corresponding range is between 0 and 30 Nm^3/thm, whereas in case of EvoDN2 models the optimal solutions are within 0 to 22 Nm^3/thm. In this case, the optimized BioGP model achieved better spread compared to the other strategies. In the same way, productivity results can also be compared, which is to be maximized. The Pareto solutions with their range and number of counts for all five years are shown in Figs. 18, 19 and 20.

cRVEA optimization process and the Pareto solutions which are generated by using this algorithm consists of four objectives and eight variables and each of these
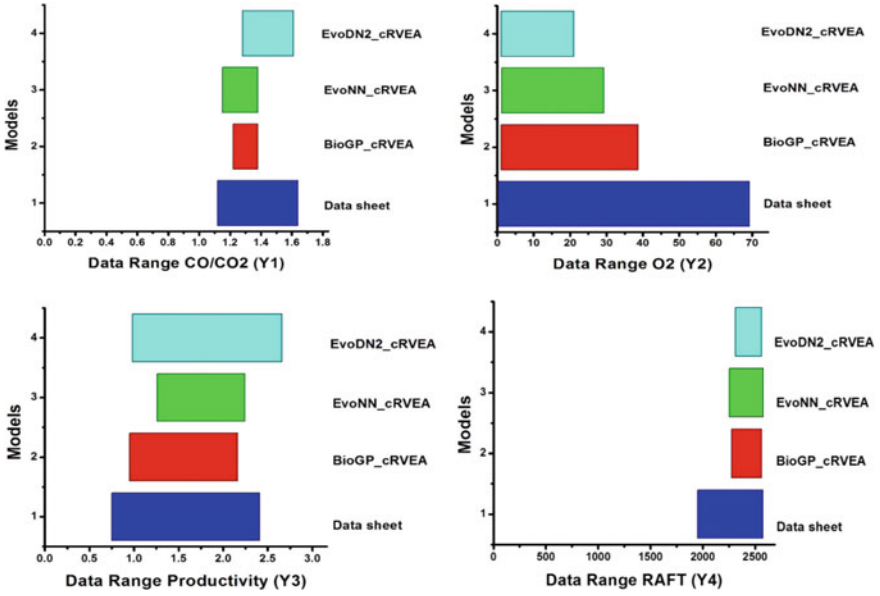
Fig. 16 Year 1 optimized solution range
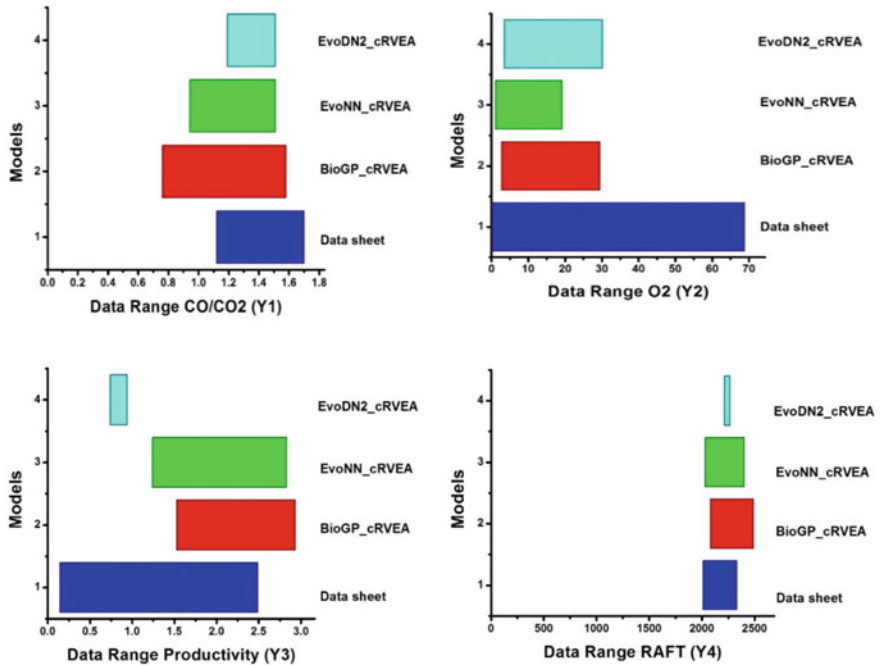
Fig. 17 Year 2 optimized solution range
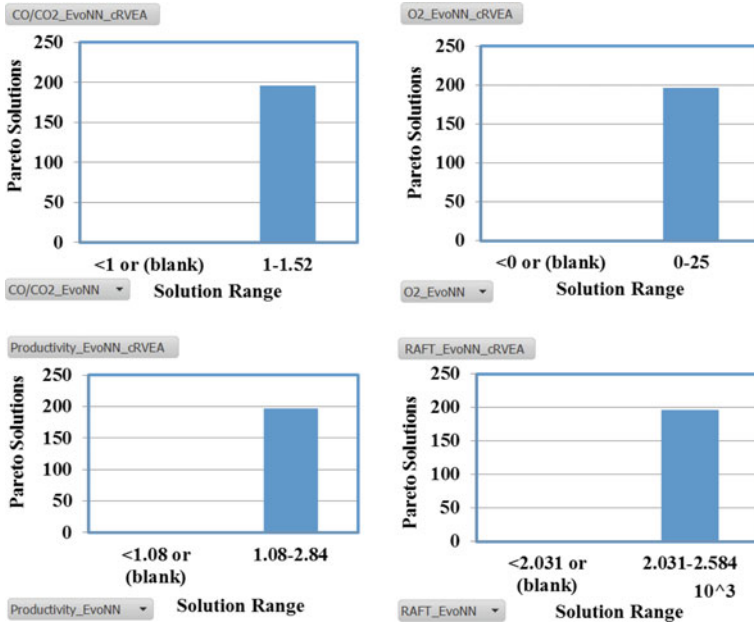
Fig. 18 Pareto solutions and range generated from EvoNN trained data by using cRVEA
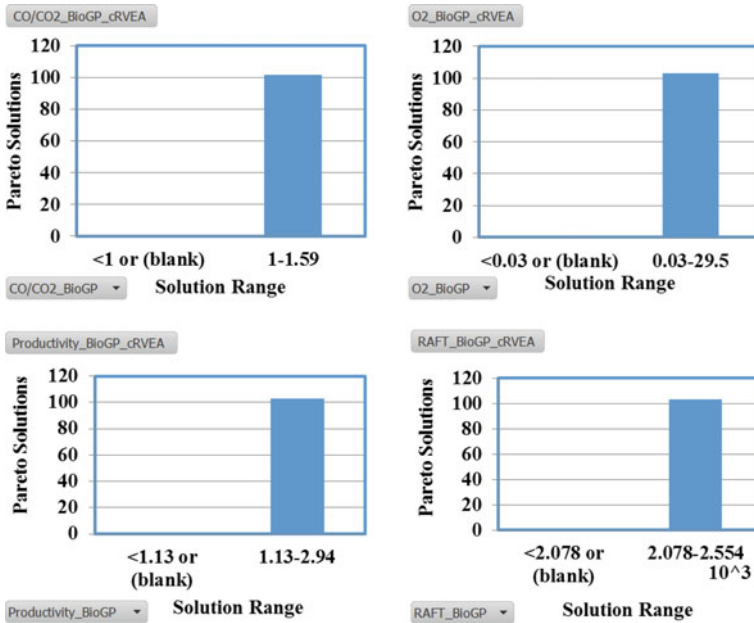
Fig. 19 Pareto solutions and range generated from BioGP trained data by using cRVEA
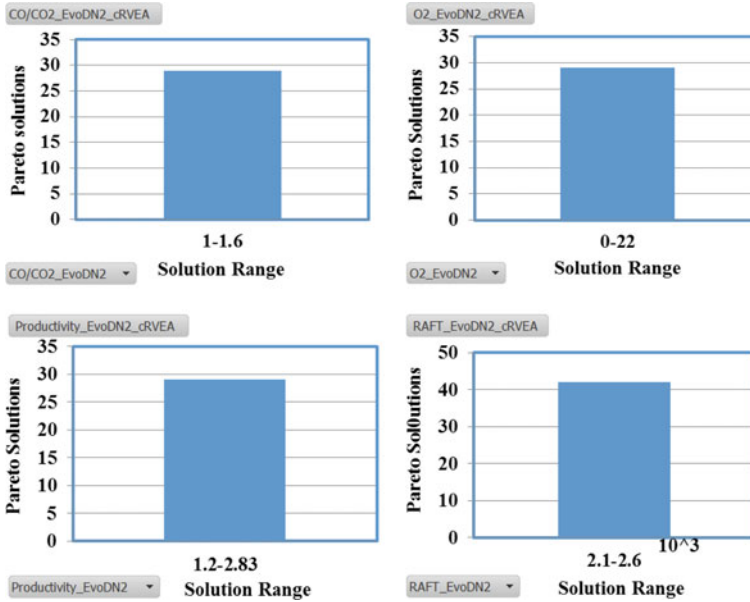
Fig. 20 Pareto solutions and range generated from EvoDN2 trained data by using cRVEA

variables are used with two additional time lags. These attributes are represented in multi-dimensional hyperspace. As per the present analysis, all the optimized objectives fall in the feasible range, which is important from the industrial operational perspective. Similarly, all the variables follow suit. The variables which are used here are organized as X11, X12, X13, X21, X22, X23, X31, X32, X33, X41, X42, X43, X44, X51, X52, X53, X61, X62, X63, X71, X72, X73, X81, X82 and X83. This is the time series representation of variable Basicity (X1), Hot Blast Volume (X2), Hot Blast Pressure (X3), Hot Blast Temperature (X4), Iron Ore (X5), Sinter (X6), Coke (X7) and Silicon (X8). The number of Pareto solutions evolved out by cRVEA algorithm consists of three series of data timelag1, timelag2 and the real-time data denoted by the number 3. After optimization all the time series data with their variables and objectives are obtained within acceptable limits, satisfying the requirement of this work. Table 8 represents the whole time series results after the optimization process and Fig. 21 presents all the Pareto solutions, which can be easily visualized in order to identify the number of solutions in the optimal set.

4.9.1 Role of Decision Making in Many-Objective Optimization Problem

The methodology suggested here provides a number of optimum solutions, out of which a Decision Maker (DM), usually a person familiar with the operational details

Table 8 Time series results after optimization process

Variable	Range	Number of optimum solutions (Y1)	Number of optimum solutions (Y2)	Number of optimum solutions (Y3)	Number of optimum solutions (Y4)
X11	0.71–1.21	302	250	192	162
X12	0.71–1.21	284	232	176	147
X13	0.71–1.21	287	235	181	152
X21	1500–2650	312	270	210	179
X22	1500–2650	263	234	185	158
X23	1500–2650	247	219	179	156
X31	1.75–2.75	265	228	180	155
X32	1.75–2.75	278	243	189	159
X33	1.75–2.75	253	226	174	149
X41	900–1200	248	196	147	123
X42	900–1200	222	172	120	95
X43	900–1200	201	150	102	81
X51	0.3–1	303	264	202	173
X52	0.3–1	302	263	204	175
X53	0.3–1	287	247	195	168
X61	0.75–1.5	268	242	194	169
X62	0.75–1.5	251	227	180	156
X63	0.75–1.5	263	228	177	151
X71	300–700	275	241	198	166
X72	300–700	231	208	170	139
X73	300–700	233	213	175	148
X81	0.25–2	320	272	213	178
X82	0.25–2	308	259	204	171
X83	0.25–2	310	263	207	175

Optimizer: cRVEA

of the blast furnace, needs to select the most suitable option, employing some additional criterion, if need be. This is to emphasize that Decision Making is an important selection process out of number of available alternatives during the course of action in a many objective scenario. Decision-makers identify the most suitable one among the solution set satisfying the constraint dictated by the environment, process and resources. In this work, the Pareto plots provide the input for decision making, by representing the solutions in a multidimensional hyperspace containing eight variables and four objectives. To a decision-maker this often is a very challenging task, as multiple numbers of attributes exist, which create difficulty for the coherent choice. For a blast furnace operator, these Pareto solutions with parallel plots can provide
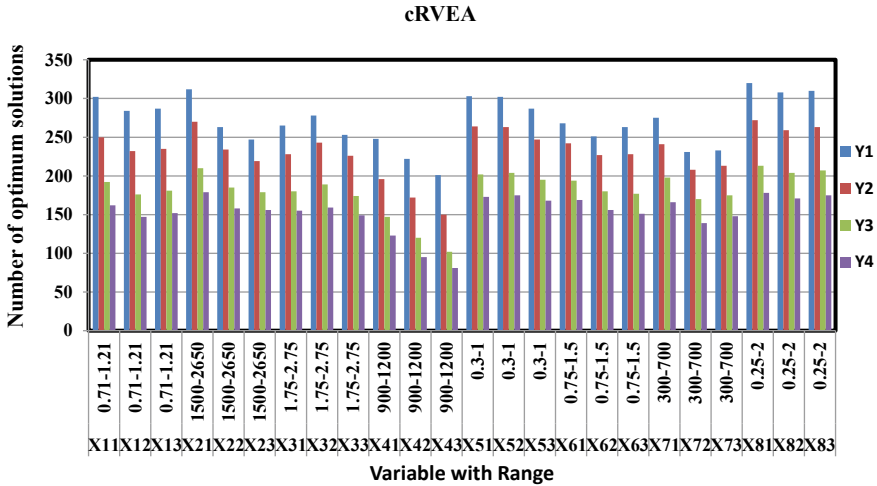
Fig. 21 Number of Pareto solutions in optimal set and the variable ranges generated by cRVEA

important source for evaluating the operational decision, as each entry in the multi-dimensional solution set is an optimum solution. A person with adequate familiarity with the blast furnace operation needs to carefully examine and analyze all the results in the parallel plots, and thereafter, according to the specific requirements of production, can come up with a recommendation or directly take the necessary action, if authorized.

5 Conclusion and Future Prospect

This chapter has discussed the implementation of data-driven modeling techniques in a unique blast furnace iron making situation, where these techniques are applied efficiently to solve a many-objective optimization problem. Various evolutionary approaches like EvoNN, BioGP and EvoDN2 are used in this work to develop the optimum data-driven models. A constraint-based evolutionary algorithm is used to access the optimum models in order to evaluate the optimal solutions for the problem. The Pareto solutions were generated within the acceptable ranges of the objective space, satisfying the regular plant operation requirements, and provided significant results considering all the used attributes. The results are significant enough so that the plant operation can be improved by periodic variation of the input parameters. The optimal solutions thus can be effectively applied in operational strategy and planning of the blast furnace iron making process. Handling a time series data of many-objective problems is a difficult work but our research clearly indicates that the metamodels constructed during this study could handle the five years nonlinear

noisy data with adequate precision and the results are computed within the acceptable objective space of the plant operation.

In future, large number of real-time problems in blast furnace iron making problem should be handled with advanced techniques. The EvoDN2 algorithm discussed here is likely to be a very useful tool for that. Improved versions of these algorithms also need to be developed in order to optimize the process at a faster rate. Some performance-based metric suitable for many-objective optimizations should be developed, which could be used with high accuracy in the hybrid algorithms to process a very large volume of data during real-time operational work. In other words, in future, similar approaches should be able to provide viable information towards in situ system control, in addition to offline decision-making in the iron and steel industry.

References

1. The White Book of Steel, *World Steel Association*, 2012. ISBN 978-2-930069-67-8.
2. Geerdes, M., Chaigneau, R., Kurunov, J., Lingiardi, O., & Ricketti, J. (2015). *Modern blast furnace iron making an introduction.* IOS Press, Delft University.
3. Omori, Y. (1987). *Blast furnace phenomenon and modeling.* London: Elsevier.
4. Ghosh, A., & Chatterjee, A. (2008). *Iron and steel making theory and practice.* Delhi: PHI learning private limited.
5. Muchi, I. (1967). Mathematical model of blast furnace. *Transaction of Iron and Steel Institute of Japan, 7,* 223–233.
6. Rist, A., & Meysson, N. (1967). A dual representation of the blast furnace mass and heat balance. *Journal of Metals, 19,* 50–59.
7. Kilpinen, A. (1988). An on line model for estimating the melting zone in a blast Furnace. *Chemical Engineering Science, 43,* 1813–1818.
8. Nath, N. K. (2002). Simulation of gas flow in blast furnace for different burden distribution and cohesive zone shape. *Material and Manufacturing Processes, 17,* 671–681.
9. Dong, X. F., Pinson, D., Zhang, S. J., Yu, A. B., & Zulli, P. (2006). Gas-powder flow in blast furnace with different shape of cohesive zone. *Applied Mathematical Modeling, 30,* 1293–1309.
10. Hatano, M., & Kurita, K. A. (1992). Mathematical model of blast furnace with radial distribution of gas flow, heat transfer and reaction considered. *Transaction of the Iron and Steel Institute of Japan, 22,* 448–456.
11. Zhou, Z., Zhu, H., Yu, A., Wright, B., Pinson, D., & Zulli, P. (2005). Discrete particle simulation of solid flow in a model blast furnace. *ISIJ International, 45,* 1828–1837.
12. Decastro, J. A., Nogami, H., & Yagi, J. (2002). Three dimensional multiphase mathematical modelling of the based on multi-fluid model. *ISIJ International, 42,* 44–52.
13. Adema, A., DEM. CFD Modelling of the Iron Making Blast Furnace. *TU Delft*, 2014, Delft University of Technology.
14. Pettersson, F., Chakraborti, N., & Saxén, H. (2007). A genetic algorithm based multi objective neural net applied to noisy blast furnace data. *Applied Soft Computing, 70,* 387–397.
15. Agrawal, A., Tiwari, U., Pettersson, F., Das, S., Saxén, H., & Chakraborti, N. (2010). Analyzing blast furnace data using evolutionary neural network and multi objective genetic algorithm. *Iron Making and Steel Making, 37,* 353–359.
16. Giri, B. K., Pettersson, F., Saxen, H., & Chakraborti, N. (2013). Genetic programming evolved through bi objective algorithms applied to a blast Furnace. *Materials and Manufacturing Processes, 28,* 776–882.

17. Mahanta, B. K., & Chakraborti, N. (2018). Evolutionary data driven modeling and multi objective optimization of noisy data set in blast furnace iron making process. *Steel Research International, 89,* 1–11.
18. Mitra, T., Pettersson, F., Saxén, H., & Chakraborti, N. (2016). Blast Furnace charging optimization using multi objective evolutionary and genetic algorithms. *Materials and Manufacturing Processes, 32,* 1179–1188.
19. Fleming, P. J., Purshouse, R. C., & Lygoe, R. J. (2005). Many-objective optimization: an engineering design perspective. *EMO, 5,* 14–32.
20. Wagner, T., Beume, N., & Naujoks, B. (2007). Pareto-, aggregation-, and indicator-based methods in many-objective optimization. *In Evolutionary multi-criterion optimization Springer Berlin/Heidelberg*, 742–756.
21. Zou, X., Chen, Y., Liu, M., & Kang, L. (2008). A new evolutionary algorithm for solving many-objective optimization problems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *38*(5), 1402–1412.
22. Chand, S., & Wagner, M. (2015). Evolutionary many-objective optimization: a quick-start guide. *Surveys in Operations Research and Management Science, 20*(2), 35–42.
23. Ishibuchi, H., Tsukamoto, N., & Nojima, Y. (2008). Evolutionary many-objective optimization: A short review. *In Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress*, 2419–2426.
24. Li, B., Li, J., Tang, K., & Yao, X. (2015). Many-objective evolutionary algorithms: A survey. *ACM Computing Surveys (CSUR), 48*(1), 13.
25. Zhang, X., Tian, Y., Cheng, R., & Jin, Y. (2016). A decision variable clustering-based evolutionary algorithm for large-scale many-objective optimization. *IEEE Transactions on Evolutionary Computation.*
26. Narukawa, K., & Rodemann, T. (2012). Examining the performance of evolutionary many-objective optimization algorithms on a real-world application. *In Genetic and Evolutionary Computing (ICGEC), 2012 Sixth International Conference (IEEE)*, 316–319.
27. Li, M., Yang, S., Liu, X., & Shen, R. (2013). A comparative study on evolutionary algorithms for many-objective optimization. *In EMO*, 261–275.
28. Ishibuchi, H., Masuda, H., Tanigaki, Y., & Nojima, Y. (2015). Modified distance calculation in generational distance and inverted generational distance. *International conference on evolutionary multi-criterion optimization,* Springer, Cham, 110–125.
29. Mahanta, B. K., & Chakraborti, N. (2020). Tri-objective optimization of noisy dataset in blast furnace iron-making process using evolutionary algorithms. *Materials and Manufacturing Processes, 35*(6), 677–686.
30. Zou, X., Chen, Y., Liu, M., & Kang, L. (200). A new evolutionary algorithm for solving many-objective optimization problems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *38*(5), 1402–1412.
31. Miettinen, K. (2012). Nonlinear multiobjective optimization. *Springer Science & Business Media*, 12.
32. Poli, R., Langdon, W. B., & Mcphee, N. F. (2008). A Field Guide to Genetic Programming. *Published Via* http://lulu.com.
33. Pettersson, F., Biswas, A., Sen, P., Saxén, H., & Chakraborti, N. (2009). Analyzing leaching data for low grade manganese ore using neural nets and multi objective genetic algorithms. *Material Manufacturing. Processes, 24,* 320–330.
34. Mondal, D. N., Sarangi, K., Petterson, F., Sen, P. K., Saxén, H., & Chakraborti, N. (2011). Cu–Zn separation by supported liquid membrane analyzed through multi objective genetic algorithms. *Hydrometallurgy, 107,* 112–123.
35. Hodge, B. M., Pettersson, F., & Chakraborti, N. (2006). Re-evaluation of the optimal operating conditions for the primary end of an integrated steel plant using multi-objective genetic algorithms and nash equilibrium. *Steel Research International, 77*(7), 459–461.
36. Chugh, T., Jin, Y., Miettinen, K., Hakanen, J., & Sindhya, K. (2016). A surrogate assisted reference vector guided evolutionary algorithm for computationally expensive many objective optimization. *IEEE Transactions on Evolutionary, 22*(1), 129–142.

37. Chowdhury, S., Chakraborti, N., & Sen, P. K. (2020). Energy optimization studies for integrated steel plant employing diverse steel-making route: Models and evolutionary algorithms-based approach. *Mineral Processing and Extractive Metallurgy Review*, 1–12.
38. Mahanta, B. K., & Chakraboti, N. (2019). Evolutionary computation in blast furnace iron making. in optimization in industry. *Springer, Cham*, 211–252.
39. Collet, P. (2007). Genetic programming in hand book of research on nature Inspared computing for economics and management. *Rennrd,J-P* (ed.), Idea: Hershey, 59–73.
40. Jha, R., Sen, P. K., & Chakraborti, N. (2014). Multi objective genetic algorithm and genetic programming models for minimizing input carbon rates in a blast furnace compared with a conventional analytic approach. *Steel Research International, 85*(2), 219–232.
41. Roy, S., & Chakraborti, N. (2020). Development of an evolutionary deep neural net for materials research. In TMS 2020 149th Annual Meeting & Exhibition Supplemental Proceedings, *Springer, Cham*, 817–828.
42. Roy, S., Saini, B. S., Chakrabarti, D., & Chakraborti, N. (2020). Mechanical properties of micro-alloyed steels studied using an evolutionary deep neural network. *Materials and Manufacturing Processes, 35*(6), 611–624.
43. Cheng, R., & Jin, Y. (2016). A reference vector guided evolutionary algorithm for many objective optimizations. *IEEE, 20,* 773–790.
44. Chugh, T., Chakraborti, N., Sindhya, K., & Jin, Y. (2017). A data driven surrogate assisted multi objective evolutionary algorithm applied to a many objective blast furnace optimization problems. *Materials and Manufacturing Processes, 32*(10), 1172–1178.
45. Li, B., Yu, S., & Lu, Q. (2003). An improved k-nearest neighbor algorithm for text categorization, arXiv preprint cs/0306099.
46. Jiang, S., Pang, G., Wu, M., & Kuang, L. (2012). An improved K-nearest-neighbor algorithm for text categorization. *Expert Systems with Applications, 39*(1), 1503–1509.
47. Qi, M., & Zhang, G. P. (2008). Trend time–series modeling and forecasting with neural networks. *IEEE Transactions on Neural Networks, 19*(5), 808–816.
48. Cortez, P., Rocha, M., & Neves, J. (2001). Genetic and evolutionary algorithms for time series forecasting. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems* Springer, Berlin, Heidelberg, 393–402.
49. Jenkins, B., & Mullinger, P. (2011). *Industrial and process furnaces: Principles, design and operation.* Elsevier.
50. Ryman, C. (2007). On the use of process integration methods: evaluation of energy and CO_2 emission strategies in blast furnace iron making and oxygen steelmaking, Doctoral *dissertation, Luleå tekniska universitet*.
51. Du, S. W., Yeh, C. P., Chen, W. H., Tsai, C. H., & Lucas, J. A. (2015). Burning characteristics of pulverized coal within blast furnace raceway at various injection operations and ways of oxygen enrichment. *Fuel, 143,* 98–106.
52. Mandal, G. K., Sau, D. C., Suchandan K. Das, & Bandyopadhyay, D. (2014). A steady state thermal and material balance model for an iron making blast furnace and its validation with operational data. *Transactions of the Indian Institute of Metals, 67*, 209–221.
53. Govindan, D., Chakraborty, S., & Chakraborti, N. (2010). Analyzing the fluid flow in continuous casting through evolutionary neural nets and multi-objective genetic algorithms. *Steel Research International, 81*(3), 197–203.
54. Chakraborti, N. (2013). Evolutionary data-driven modeling. In Informatics for materials science and engineering, *Butterworth-Heinemann*, 71–95.
55. Li, M., Zhen, L., & Yao, X. (2017). How to read many objective solutions sets in parallel co-ordinates. *School of computer science*, University of Birmingham, UK.

A Brief Appraisal of Machine Learning in Industrial Sensing Probes

R. Biswas

Abstract Machine learning has come a long way since its inception. It has totally revolutionized the industrial scenario. For decades, there is clear evidence of the usage of sophisticated digital control and monitoring systems by industrial operators. These entail a multitude of sensors with defined functionalities. Through the adoption of machine learning, their handling becomes a lot easier. In this chapter, the different strategies adopted through machine learning are being highlighted. Additionally, the challenges of effective integration of industrial data, including that from sensors, for standard ML are outlined. Apart from this, this chapter appraises readers about predictive maintenance; accompanied by recommendations.

1 Introduction

The world population is growing at an alarming rate. As per the world meter, the population of the world stands at 7.8 billion people as of now. With the rising population day by day, the demand for products is also mounting. The growth rate will always be escalating without any sign of being stagnant. Consequently, the need for essential products is directly proportional to no. of consumers. Now, to meet the demand for products, we have to be heavily dependent on pieces of machinery related to manufacturing. With heavy pieces of machinery, it is required to deploy a workforce for control and operation. Moreover, heavy pieces of machinery are often linked to manufacturing at small-scales. Regardless of their dimensions, there should be regular maintenance of the pieces of machinery, thereby enabling timely

R. Biswas (✉)

Applied Optics and Photonics Lab, Department of Physics, Tezpur University, Tezpur, Assam, India

e-mail: rajib@tezu.ernet.in

production as well as processing. Now, the main issue arises. If the no. of components increases, then the complexity also increases thereby. The situation becomes clumsier if the no. of assorted sensors/probes mounts too.

The industrial processes span a wide range of constitutional components. From ground level production to delivery of products, there exist several underlying steps. For example, if we have a food production unit, we have to sequentially undergo the processing of raw materials to the final packaging of goods. However, in between, it is essential to have regular maintenance of cleaning the benchtop rigs. This cleaning part is itself a tedious job that can be done manually or through an assembly of assorted probes. In case of manual cleaning, it is necessary to dismantle all the components, followed by the painstaking assembling of them. Accordingly, dedicated sensors are fitted in for cleaning purposes. Likewise, in the case of chemical industries dealing with the gas–liquid mixture, it is very much essential to have a close eye on proper mixing via allied sensory schemes so that desired conditions can be attained. All these instances converge to one point that industrial sensor probes relying on different methods are part and parcel of the system. Meanwhile, these different sensor methods exhibit a wide range of diversity so far in their operating factors, namely, speed of data access and monitoring coverage of the spatial area. It is needless to mention that a single method for managing these sensors is meaningless. Equally important is the fact that we have to explore appropriate data analysis methods so that the recorded measurement to the system under inspection can be effectively linked. With a growing no. of nodal points (i.e., sensors), there emerges voluminous data which happen to be multidimensional. At this juncture, individual monitoring of all data in order to perceive the health of machinery is a herculean task. Meanwhile, if the overall production system is undergoing a gradual decline leading to malfunction, there should be a mechanism to overhaul it and stop it.

To deal with such exacerbating conditions, machine learning appears as a savior [1–11]. Machine learning has proven to be a robust tool for data analysis. Of late, it has become an integral part of an industry. Meanwhile, the advent of technological developments leads to overwhelming demand for cost-effective production. As a result, there is a paradigm shift towards the automation of components related to production. At the same time, the quality of output becomes another concern. Precisely, in the long run of achieving ultra-automation with simultaneous optimal cost-reduction, industrial artificial intelligence and machine learning emerge as rescuer [6–16]. They can be regarded as one of the driving factors, which add extra impetus to ultra-automation [17–20]. This chapter gives an overview of the modus operandi of machine learning as related to industrial probes. It highlights several key issues of ML related to data preparation to the management of it to achieve proper functioning. Additionally, predictive maintenance through ML is comprehensively outlined which is further accompanied by future recommendations.

2 Functional Flowchart of ML

With the ready availability of data along with the provision of ease storage and capture, AI and ML provide ample scope of utilizing data in order to enhance production yield, thereby ensuring quality and better security of involved personnel. In such cases, manufacturers devise the ML, aided by AI in a systematic manner. First, there are several industrial probes/sensors, which produce a copious amount of data. In order to implement ML, we have to proceed through certain steps. The first step is the pretreatment of data. After that, we have to devise feature extraction/pattern recognition and dimension reduction. It is then followed by system modeling (like predictive maintenance). Figure 1 illustrates the flow chart of Industrial ML [18–20].

2.1 Data Pretreatment

In general, industrial pieces of machinery are equipped with a multitude of sensors or probes. In order to build up an effective ML approach, it is imperative that the colossal data generated from sensors should be polished through a data pretreatment. Accordingly, we can adopt some of the data pretreatment processes like noise removal, removal of baseline, outlying data, as well as normalization of data [3–9]. Each of these processes is necessary in order to streamline machine learning. We elaborate on each of them which are as follows [3–9].

2.1.1 Erasing Noise

It is a common fact that several sensors pertaining to the industry will generate a huge data set. As such, there is no doubt that there will be a proliferation of noise. Again, time series forms the main constituent of sensor data. Consequently, it is bound to be contaminated by noise. In order to erase noise, there are several techniques. We can cite the Fourier analysis and the wavelet analysis. Wavelet is a little bit advantageous
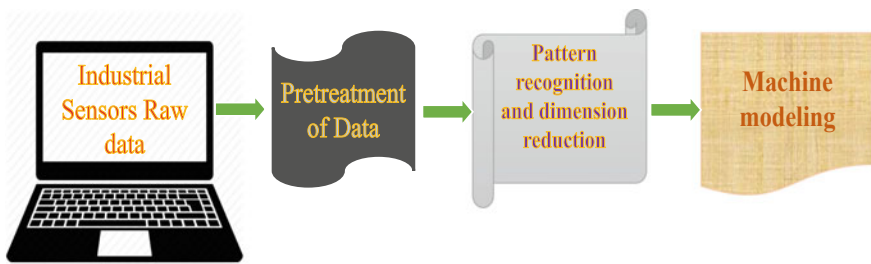
Fig. 1 Workflow of machine learning corresponding to industrial probes

as it handles data both in time and in frequency domain. As a result, the time series can be split into high and low frequency domains. Often, the noise accompanies the high-frequency domain. In such a case, the signal can be filtered as well as enhanced if we set a threshold level in the high-frequency part through alteration of the wavelet coefficients in the same domain. Additionally, one can also resort to using Autoregressive analysis [6–10].

2.1.2 Removing Baseline

Removing the baseline is another effective way of pretreating the data. Often, the signal is not properly aligned exhibiting quite chaotic characteristics. This leads to false data representation as well as misinterpretation. To do away with it, the spectrum is inspected for baseline key points, followed by the construction of the baseline model. Once it is through, the baseline is subtracted from the time series, resulting in a corrected and proportionate signal. The adaptive learning algorithm is one of the widely used techniques for achieving it [10–12].

2.1.3 Outlier Sensing

Noise eradication is not the end in managing data yielded by sensors. Equally important is the proper identification of outliers prevalent in sensor data. When they are present, they create difficulty in precise sensory analysis. To begin with, outliers refer to anomaly points or observations not fitting in the trend of data, or we can refer to doubtful observations. If not identified properly, they can lead to erroneous modeling of systems. There are widely accepted statistical tools like regression analysis; wherein, a standardized residual value is kept as a comparative scale. Moreover, a self-organization map, which is based on the separation of data points, is another tool that can be used in nonlinear systems. However, we have to use it with caution. Sometimes, certain data points exhibit considerable deviation from the majority, thus surpassing the outliers too. To deal with such complex nonlinear systems, we can use radial basis function neural networks (RBFNN). It is worthwhile to cite that exclusion of outliers from the analysis may hinder us to get insight into hidden characteristics and operation behavior, as outliers are different from noise [13–16].

2.1.4 Normalizing Data

The last step for pretreatment of data is normalization, which can be divided into two categories, namely, local and global. While local normalization refers to unit scaling, global normalization usually proceeds with the association of relative sensor variables. The idea behind this is that it does not allow the variables of smaller magnitudes to be overshadowed by the ones with higher magnitudes. This also may

amplify noise or irrelevant information inherent in sensor data that may prove to be tricky for overall system modeling [14–17].

2.2 Pattern Recognition and Dimension Reduction

Once the data is pretreated, there arises another vital component for ML. It is none other than feature extraction which paves way for downsizing the dimension. By this time, we have learned that industrial pieces of machinery are associated with a large no. of probes. These probes or sensors produce voluminous data. With the aid of modern developments, there comes a lot of novel sensors that are engaged in complex industrial pieces of machinery. The data generated by them are of high quality as well as precision. Since various sensors are engaged, hence, data produced will be of multidimensional and in large capacity. Hence, it becomes imperative to analyze these colossal data through certain specialized techniques. Among them, feature extraction and dimension reduction are one of the primarily used ones. As the name goes, it identifies definite patterns, and subsequent to pattern recognition, the analysis tool then proceeds to reduce the dimensions. For example, if we have p dimensions as the output by sensor data, then we map it to another reduced dimension of q through the extraction of distinctive patterns [17–20]. Thus, we can write $f: M_p \rightarrow M_q$ ($p \gg q$). Due to the adoption of this unique step, there occurs a reduction in substantial computational loads from the next step of system modeling. Meanwhile, we can also keep noise and redundant data at abeyance through this step. In order to visualize this step, we can mention several algorithms such as multivariate analysis and principal component analysis.

2.3 Machine Modeling

As we know, every manufacturing operation is futile if proper maintenance is not taken care of timely. It so happens that some utility functions of a manufacturing plant may go out of order due to lack of maintenance. In such cases, there is an imminent need for a mechanism, which can at least predict the non-functionality or lag in operation. Accordingly, there pops up the term predictive maintenance as an integral part of machine modeling. This can be regarded as a total game-changer as operations' cost/expenditure can come down to a considerable level. Earlier, it has been visualized through supervisor control and data acquisition (SCADA). Precisely, SCADA refers to control system architecture. Being composed of computers, networked data communications, and graphical user interfaces (GUI), it plays a vital role in supervising as well as maintaining programmable logic controllers (PLC) and discrete proportional-integral-derivative (PID) controllers. This has a direct link with plants

or machinery. Meanwhile, SCADA also translates to operations related to construction processes, which are solely project-driven. If we look at the operational procedure of SCADA, there are certain restrictions. As it contains a couple of levels in its architecture, hence, there is every possibility of data lag/communication gap among the levels. Apart from that, the supervisory workflows are mostly static role-based. As a result, it cannot directly cater to a dynamic system. In other words, we can term SCADA as semi-manual. As such, the machine behavior is partly fixed thus giving a little scope of anomaly detection. Another noteworthy point is that pieces of machinery are based on patterns, which exhibit dynamic behavior. This kind of altering pattern remains unnoticed in case of SCADA. Moreover, contextual data pertaining to the manufacturing process are too large to be handled by SCADA. Let us take an example. Let us suppose that one of the assorted sensors of a production unit is suddenly sensing an abnormal rise in temperature. When the supervisory approach is static rule-based, it cannot identify the actual problem. Being oblivious of the undergoing sterilization of the machinery, SCADA may ring an alarm that may be a false-positive case. In such situations, we can have the utility of ML. ML algorithms thrive to make synchrony between machinery and production flow. To make it happen, there must be an input of Operational Technology (OT) as well as Information Technology (IT) data. The OT spans data from the production floor, sensors, Programmable Logic Controllers, SCADA along with historians, whereas IT includes Enterprise Resource Planning, behaviors, quality which is further accompanied by Machinery Execution system data. With a convergence of OT and IT data, ML can assist in proper maintenance. Now, when we move over to Industrial Artificial Intelligence, the picture becomes more sophisticated. The extra addition in terms of "Training" makes ML capable of sensing anomalies. Further, it aids in the testing of correlations for identifying patterns from a large feed of data. Being endowed with the ability of handing analysis of a large amount of data in real time, it renders doable retorts corresponding to any impending issues. While doing so, every asset and system is under constant scrutiny, which results in the prior diagnosis of failure or malfunction. Being aided by digital twins, ML can offer insights for further action. Here, digital twins refer to replicas of physical devices in virtual space with the sole aim of running simulations before deployment of the actual type. As such, they work in synergy with ML and AI [19, 20].

Enabling Predictive Quality Analytics with Machine Learning

Downtime prevention cannot be considered as the ultimate objective of AI. Maintaining the quality of yield/output is another top priority. ML can assist us in the prognosis of quality deterioration. Accordingly, the raw materials' wastage can be avoided in case there is an indication of downscaling of products. It thereby saves valuable time invested in the production of inferior products.

We can divide ML into two main categories, namely, supervised and unsupervised learning. The whole process is depicted in Fig. 2.
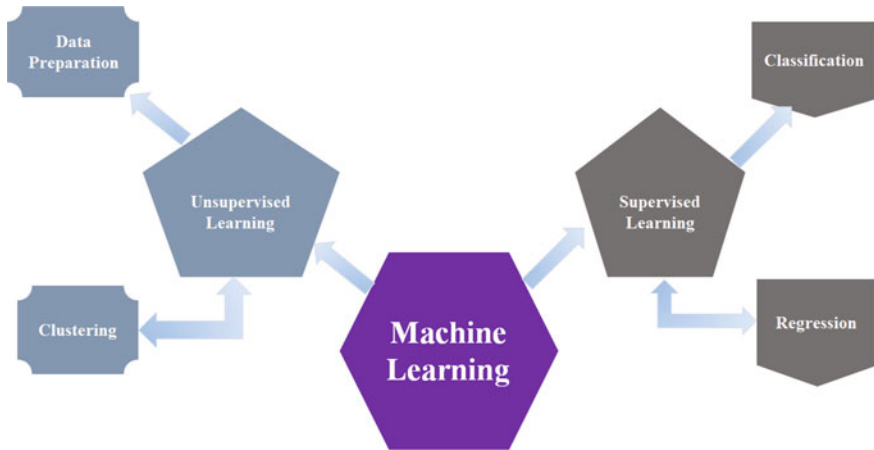
Fig. 2 Machine learning tree for predictive maintenance

2.3.1 Supervised Machine Learning

This is treated as one of the most widely used techniques pertaining to the manufacturing process. It usually leads one to a predefined target. Suppose, we feed input, and thereby, we want to have the output. Now, our main work would be to explore the mapping function that would link these two variables. As such, supervised ML necessitates the proliferation of many D's, namely, data input, data training and defining, and data visualizations through selective algorithms. The main idea behind this is to construct a mapping function at a prescribed accuracy level, thus bestowing upon the ability of predicting outputs upon arrival of new inputs in the system. The schematic of supervised ML is illustrated in Fig. 3.

To start with, the selective algorithm is entered via a training dataset. Then, after several iterations, it is made to achieve its defined output. Meanwhile, when the desired level of learning is attained, the learning process ceases. As discussed in the previous section, it is pointed out that supervised ML aids predictive maintenance which is of utmost importance in manufacturing. This can be accomplished through two ways, namely, classification and regression. Although, accomplishment is executed at different levels, yet, both these processes own the identical goal of mapping the function connecting input and output data. When we refer to input data, we mean data related to the manufacturing process. On the other hand, output data spans part failure, overheating, time lag, etc., as known possible results.

(a) **Classification**

The classification relates special attributes to a well-defined dataset. Although it is confined to a Boolean value response, it proves very beneficial in achieving accuracy to a superior level. For instance, we can cite classifications attributed to email filters, which demarcate spam emails from useful ones. To execute classification, we can
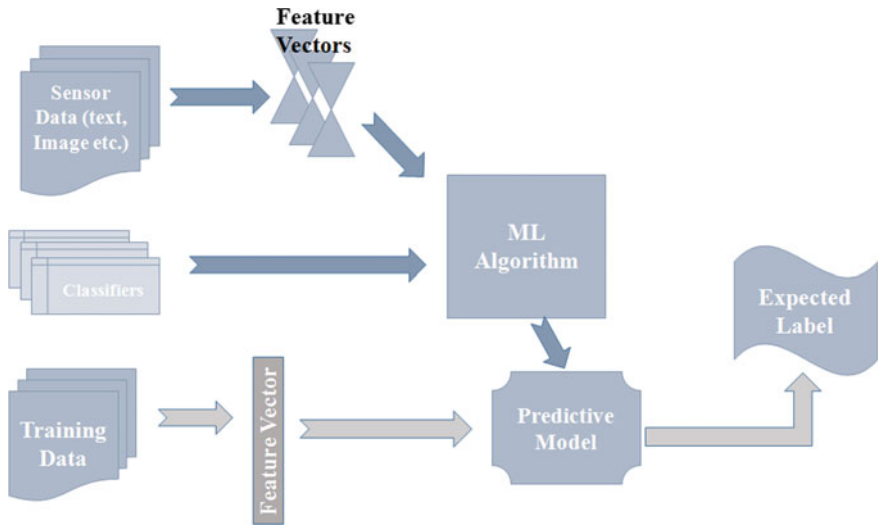
Fig. 3 Schematic of supervised machine learning

adopt algorithms comprising logistic regression and artificial neural networks. In case of industrial sector, predictive maintenance contains a multitude of classifications, as it has to deal with several plausible causes of machine failure, crack detection, as well as malfunctions of components as alerts from allied sensors. In such scenario, multi-class classifications are applied for these possible outcomes outlining potential issues related to equipment. Towards this goal, it is evaluated based on several variables incorporating machine health, malfunction, etc.

(b)   **Regression**

Regression is one of the other ways through which ML-based predictive maintenance is performed. It is applicable when there exists data lying within a range. Regression is used when data exists within a range collected from sensors. Regression helps in evaluating one important factor, i.e., Remaining Useful Life of an asset. The asset here implies any component or assembly of the machinery. Through RUL, we can have a priori information of the duration of the lasting of the next component/assembly. In general, linear regression is adopted as the commonly used machine-learning algorithm, yielding easily interpretable output. As an example for regression-based activity, we can cite the example of fouling removal of food material from benchtop rig. This fouling emerges as one of the obstacles for cleaning of equipment with a view to maintaining hygiene in food production. We can adopt a neural network regression model to forecast the fouling remaining on the surface which can be subsequently eliminated through effective cleaning mechanisms.

2.3.2 Unsupervised Machine Learning

As the name goes, this ML process is very different from the earlier one in the sense that it does not own any predefined/expected outcome. In the earlier ML, we have the liberty to train the algorithm in tune with the expected outcome. However, when we are quite uncertain about the outcome, one can resort to unsupervised ML. To improvise unsupervised ML, we can implement clustering and data preparation as well.

(a) **Clustering**

It so happens that data scientists are in a dilemma when the source of information depicting data is undefined. Apart from this, the outcome is already unknown. In such cases, clustering can be brought to salvage this situation. Through the formation of clusters of common data sharing definite attributes, it becomes possible for ML to explore the hidden patterns. Additionally, another wonderful feature of clustering is that it helps in diminishing unwanted portions. More appropriately, we can term it noise, which implicates irrelevant parameters lying within data while we handle colossal data variables.

(b) **Data Preparation**

Now we enter into another important domain of predictive maintenance. So far, we have understood that ML is strongly dependent on data. Type of data as well as the quality remains the integral part of ML, thereby ensuring correct turnouts to end-users. Let us simplify the discussion by taking the example of machine failure as monitored by predictive maintenance. When we initiate working on failure events, we have to capitalize on earlier data about the performance of the machinery, further to be complemented by maintenance records. All sorts of these historical data sets will enable us to foresee any events due in the future, corresponding to the machinery.

As we know, a production machine can last a number of years depending on use & maintenance. This results in historical data for that duration. Now, the main concern creeps in. That historical data should at least date back to the period where there is evidence of the deterioration process of the machine. Once we gather this information, we must have in possession of other static information. By other static information, we mean characteristics of the machine, mechanical properties, usual utility properties as well as environmental operating conditions. After accruing these, we have to cover up the important aspects, which play a decisive role in devising out the strategy for predictive maintenance. Accordingly, some key aspects that need urgent attention are as follows:

(a) Identification of failures related to a machine or system.
(b) Selection of failure events that require prediction.
(c) Characterization of the failure events regarding their occurrence like a sudden, steady decline.
(d) Diagnosing the components connected to the failure events.

(e)   Measurable parameters signifying the state of component or machine health.
(f)   Affixing the accuracy and frequency of measurements.

Considering these concerns, we can see that there has to be a synergy between domain specialists and data scientists.

3   Survey

In the previous sections, we discuss the various ML algorithms in detail, which provide a basis for predictive maintenance. Let us shift our attention to the current trend that is adopted by researchers as well as the practices adopted by the industries. Accordingly, in Table 1, we enlisted many recent works executing different sorts of tasks by machine learning through the adoption of diverse algorithms. The Fig. 4 depicts one schematic illustration of industrial probes for adaptation of ML. For instance, Forte et al. [21] developed a methodology in order to monitor the gas–liquid mixing regime within gas–liquid and gas–solid–liquid mixtures. For that purpose, they accrued data from the piezoelectric sensor from acoustic emission. Marinating

Table 1   ML aided industrial probes

Sl. no.	Industrial probe	Machine learning algorithms	Accuracy	References
1	Acoustic piezoelectric sensor	logistic regression, support vector machine (SVM), k-nearest neighbor (k-NN) and decision tree	90% (logistic regression)	[21]
2	Ultrasonic and optical sensor	Logistic regression	97–98% in prediction	[22]
3	LOS/NLOS links	Supervised classification	–	[23]
4	Temperature, pressure and vibrations probes	Support Vector Machine (SVM) and the Multilayer Perceptron (MLP)	98.1	[24]
5	Eddy current probe	Quantitative image analyzing	–	[25]
6	CMOS image sensor	Random Forest model	–	[26]
7	Ultrasonic sensors	Regression	96.1	[27]
8	Drilling sensors	Principal component analysis		[28]
9	Ultrasonic probes	Classifiers	100%	[29]
10		Deep neural network		[30]
11	Long range ultrasonic transducers	Support Vector Machine classifiers	–	[31]
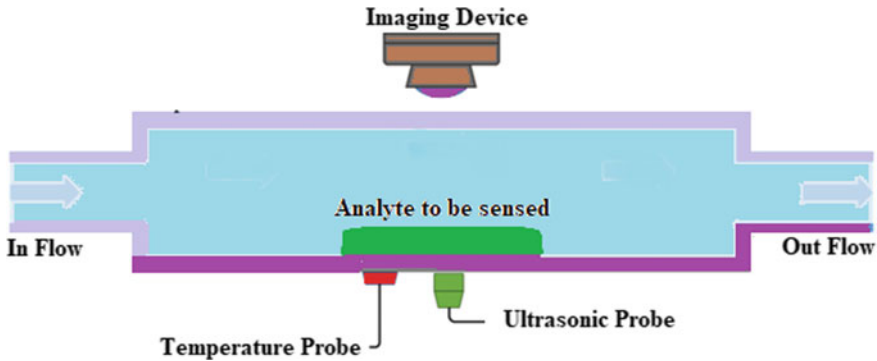
Fig. 4 Schematic illustration of industrial probes for adapting ML

the analysis in three different flow regimes, they were successful in achieving an accuracy level of 90% of the operating regime both in the presence and in the absence of suspended particles in the mixture.

In another work, Simeone et al. [22] had reported an effective monitoring and food cleaning mechanism. As we know, proper cleanliness is mandatory for keeping hygiene in the food industry. However, it necessitates a huge amount of water and other essentials. In order to optimize such processes, Simeone et al. used optical and ultrasonic probes to produce tailored signals and image processing to effectively run the cleaning process. Via the neural network regression model, they were able to predict area and volume with an accuracy of 98% and 97%, respectively. Again, wireless probes and actuators are part and parcel in establishing an indusial IoT. However, the varying conditions result in a loss of transmission among the nodes. Meanwhile, there arise uncontrollable disparities in radio-link in the case of line of sight as well as non-line of sight. To cope up with this problem, Bombino et al. [23] executed a supervised ML algorithm with variable complexity for the inference of the radio-link state by engaging the effects of the limited sampling frequency, bit-depth, and moving average filtering. Thereby, they were able to attain a classification accuracy of LoS/NLoS radio links successfully. In another feat, Orrù et al. [24] reported a simple and easy-to-implement machine learning (ML) model for early fault prediction of a centrifugal pump in the oil and gas industry. They pointed out the scheme to be inexpensive and reliable. Using two competing ML algorithms viz. SVM and multilayer perception, they gathered data from temperature, pressure, and vibration probes and thereby managed to identify potential faults with high precision, thus leading to fault prediction alerts. Zhang et al. [25] disclosed an eddy current (EC) testing probe where they used coils carrying three-phase currents as excitation, and for sensing, they utilized integrated array tunnel magnetoresistance (TMR) sensors measuring the magnetic field as a receiver. Via quantitative image analyzing aided by ML, they were able to show excellent sensitivity in detecting defects in horizontal as well as vertical alignments. Additionally, the adaptation of an artificial neural network helped them to predict defects located within the 1 mm range. In another

work, Hussain et al. [26] devised a particle analyzer, solely based on light scattering. They implemented a collimated beam configuration using a consumer electronic camera and machine learning. By using a small form factor angular spatial filter, they accrued high-resolution images that are improvised with ML. This approach led them to accurately predict the volume median diameter; thus paving way for use outside a standard laboratory, for example, in online and in-line industrial process monitoring. Meanwhile, Bowler and coworkers [27] developed a machine learning procedure by acquiring and processing signals related to thrust force, torque, acoustic emission, and vibration during drilling. It was actually a multisensory approach. Through robust use of principal component analysis as well as feature extraction, they managed to accurately diagnose tool flank wear. In another field of mixing, Caggiano et al. [28] adopted ultrasonic sensors to predict mixing states. With a synergetic combination of SVM and regression classifiers, they were able to decipher four-level mixings as well as time remaining before mixing was fully achieved. As per their report, there happen to be classification accuracies of up to 96.3% for honey-water blending and 92.5% for flour-water batter mixing. Simultaneously, the R2 values for both the regression models were found to be remarkable. In another work by Escrig et al., it had been shown that food and drink production equipment could be routinely monitored via the use of ultrasonic measurements and a range of different machine learning classifications. It would be possible to come up with a prediction of fouling material in plastic and metal pipes nearly to an accuracy of 100%. The work actually could be considered as a boon in reducing maintenance expenditure. Munir et al. [30] demonstrated a data-driven neural network approach. They showed that defects in could be classified to a high accuracy through the adoption of ultrasonic probes where there is no prior hand need for feature extraction. Similarly, failure prediction in the oil and pipe industries is very important. If not handled carefully, it can cause disruptions from production to allied sectors. To implement a full-proof method, Lee et al. [31] demonstrated a failure prediction system for oil and gas pipelines. As probes, they used long-range ultrasonic transducers. Via Euclidean-Support Vector Machines classification approach, they were able to show a mechanism of making a decision on the integrity of the pipeline in a continuous monitoring environment. As per their report, the classification accuracy of SVM is independent of the kernel function and the data emanating from the pipes and defects simulation. All these works are summarized in Table 1.

4 Final Remarks and Recommendations

It is quite evident that AI and ML in the industry have produced considerable beneficial changes, which eventually lead to augmented efficiency. Accordingly, it paves way for new business opportunities, accompanied by the opening of more avenues. It is worthy to mention that when ML is equipped with IoT, this enables manufacturers to ease the productivity as well as reduce the expenditures associated with maintenance. The supervised models can be exploited to acquire perceptions from

the data. The hitherto obtained information assists in prognosis as well as prediction. The prediction not only ensures a smooth and proficient manufacturing process but also minimizes cost related to maintenance and forbids degradation in the quality of outputs. It is also important to remember that ML techniques thrive to explore hidden features, which are implicit in complex systems. These complex systems are often deceptive in nature along with elusive physical laws. As a result, ML should have a continuous self-adjusting agenda with cognitive capability; albeit, ML is escorted by AI and advanced statistical algorithms. We can emphasize that there must be the existence of a validating process and a possible model parameter updating process. Equally important are some other issues such as adaptability, training period, threshold level, and accuracy setting for robust ML processes. As we cannot rule out the possibility of interference among them, hence, there should be synergized integration to yield an effective ML for sensor/probe for the industry. If we can resort to the deployment of ML models, which are transferable in nature, it would be a big boon for industrial probe design as we can nullify the possibility of the whole reorientation of a new environment or system. Developing transferrable machine learning models is another attractive direction for novel sensor design, as it can prevent a complete reorientation for a new system or environment. However, all these have to be complemented by novel algorithms, which could result in improved sensor analysis for effecting a healthy ML environment.

References

1. Bishop, C. M. (2006). *Pattern recognition and machine learning*. New York: Springer.
2. Berrueta, L. A., Alonso-Salces, R. M., & Heberger, K. (2007). Supervised pattern recognition in foodanalysis. *Journal of Chromatography A*, *1158*, 196–214.
3. Addison, P. S. (2002). *The illustrated wavelet transform handbook—Introductory theory and applications in science, engineering, medicine, and finance*. London: Institute of Physics Publishing.
4. Cappadona, S., Levander, F., Jansson, M., James, P., Cerutti, S., & Pattini, L. (2008). Wavelet-basedmethod for noise characterization and rejection in high-performance liquid chromatographycoupled to mass spectrometry. *Analytical Chemistry, 80,* 4960–4968.
5. Krebs, M. D., Tingley, R. D., Zeskind, J. E., Kang, J. M., Holmboe, M. E., & Davis, C. E. (2005). Autoregressive modeling of analytical sensor data can yield classifiers in the predictor coefficient parameter space. *Bioinformatics, 21,* 1325–1331.
6. Ubeyli, E. D., & Guler, I. (2004). Spectral analysis of internal carotid arterial Doppler signals using FFT, AR, MA, and ARMA methods. *Computers in Biology and Medicine, 34,* 293–306.
7. Pearson, G. A. (1977). General baseline-recognition and baseline-flattening algorithm. *Journal of Magnetic Resonance, 27,* 265–272.
8. Esposito, A., & D'Andria, P. (2003). An adaptive learning algorithm for ECG noise and baseline driftremoval. *Neural Nets, 2859,* 139–147.
9. Shusterman, V., Shah, S. I., Beigel, A., & Anderson, K. P. (2000). Enhancing the precision of ECG baseline correction: Selective filtering and removal of residual error. *Computers and Biomedical Research, 33*, 144–160. Algorithms 2008, 1, 148
10. Krebs, M. D., Tingley, R. D., Zeskind, J. E., Holmboe, M. E., Kang, J. M., & Davis, C. E. (2006). Alignment of gas chromatography-mass spectrometry data by landmark selection from complex chemical mixtures. *Chemometrics and Intelligent Laboratory Systems, 81,* 74–81.

11. Krebs, M. D., Kang, J. M., Cohen, S. J., Lozow, J. B., Tingley, R. D., & Davis, C. E. (2006). Two dimensional alignment of differential mobility spectrometer data. *Sensors and Actuators B, 119,* 475–482.
12. Crowe, C. M. (1989). Test of maximum power for detection of gross errors in process constraints. *Aiche Journal, 35*, 869–872.
13. Mah R. S. H., & Tamhane, A. C. (1982). Detection of gross errors in process data. *Aiche Journal, 28*, 828–830.
14. Prescott, P. (1975). Approximate test for outliers in linear-models. *Technometrics, 17,* 129–132.
15. Munoz, A., & Muruzabal, J. (1998). Self-organizing maps for outlier detection. *Neurocomputing, 18,* 33–60.
16. Zhao, W. X., Chen, D. Z., & Hu, S. X. (2004). Detection of outlier and a robust BP algorithm againstoutlier. *Computers & Chemical Engineering, 28,* 1403–1408.
17. Wold, S., Esbensen, K., & Geladi, P. (1987). Principal component analysis. *Chemometrics and Intelligent Laboratory Systems, 2.*
18. file:///E:/Book%20Chapter%20on%20Ml/Machine%20Learning%20and%20AI%20in%20 Manufacturing%20-%20The%20Complete%20Guide.html. Retrieved August 1, 2020.
19. Moraru, A., Pesko, M., Porcius, M., Fortuna, C., & Mladenic, D. (2010). Using machine learning on sensor data. *Journal of Computing and Information Technology—CIT, 18*(4), 341–347.
20. Zhao, W., Bhushan, A., Santamaria, A. D., Simon, M. G., & Davis, C. E. (2008). Machine learning: A crucial tool for sensor design. *Algorithms, 1,* 130–152. https://doi.org/10.3390/a10 20130.
21. Forte, G., Alberini, F., Simmons, M., & Stitt, H. E. Use of acoustic emission in combination with machine learning: Monitoring of gas–liquid mixing in stirred tanks. *Journal of Intelligent Manufacturing.* https://doi.org/10.1007/s10845-020-01611-z
22. Simeone, A., Woolley, E., Escrig, J., & Watson, N.J. Intelligent industrial cleaning: A multi-sensor approach utilising machine learning-based regression
23. Bombino, A., Grimaldi, S., Mahmood, A., & Gidlund, M. (2020). Machine learning-aided classification of LoS/NLoS radio links in industrial IoT. *IEEE Explore*
24. Orrù, P. F., Zoccheddu, A., Sassu, L., Mattia, C., Cozza, R., & Arena, S. A. (2020). Machine learning approach using MLP and SVM algorithms for the fault prediction of a centrifugal pump in the oil and gas industry. *Sustainability, 12,* 4776. https://doi.org/10.3390/su12114776.
25. Zhang, N., Ye, C., Peng, L., & Tao, Y. Eddy current probe with three-phase excitation and integrated array TMR sensors. *IEEE Transactions on Industrial Electronics*. https://doi.org/10. 1109/tie.2020.2989704.
26. Hussain, et al. (2020). Ultra-compact particle size analyzer using a CMOS image sensor and machine learning light. *Science & Applications, 9,* 21.
27. Bowler, A. L., Bakalis, S., & Watson, N. J. (1813). Monitoring mixing processes using ultrasonic sensors and machine learning. *Sensors, 2020,* 20. https://doi.org/10.3390/s20071813.
28. Caggiano, A., Angelone, R., Napolitano, F., Nele, L., & Teti, R. (2018). Dimensionality reduction of sensorial features by principal component analysis for ANN machine learning in tool condition monitoring of CFRP drilling. *Procedia CIRP, 78,* 307–312.
29. Escrig, J., Woolley, E., Simeone, A., & Watson, N. J. (2020). Monitoring the cleaning of food fouling in pipes using ultrasonic measurements and machine learning. *Food Control.*
30. Munir, N., Kim, H. J., Song, S. J., & Kang, S. S. (2018). Investigation of deep neural network with drop out for ultrasonic flaw classification in weldments. *Journal of Mechanical Science and Technology, 32,* 3073–3080.
31. Lee, L. H., Rajkumar, R., Lo, L. H., Wan, C. H., & Isa, D. (2013). Oil and gas pipeline failure prediction system using long range ultrasonic transducers and euclidean-support vector machines classification approach. *Expert Systems with Applications, 40,* 1925–1934.

Mining the Genesis of Sliver Defects Through Rough and Fuzzy Set Theories

Itishree Mohanty, Partha Dey, and Shubhabrata Datta

Abstract The actual cause of sliver defect is difficult to determine, since the defect usually reveals itself after the rolling process (hot/cold) is complete. The genesis of sliver defect in cold rolled steel sheets is investigated in this work using two popular computational intelligence tools used in data mining, namely, rough set and fuzzy set theories. A substantial amount of data starting from the steelmaking stage to finish rolling of the product has been collected with the aim of extracting useful knowledge about plausible cause(s) of sliver formation. While rough set theory helps to select the important variables to which the cause of the defect can be attributed in the form of rules, these rules are given a linguistic form through fuzzy membership functions. A rule base thus evolves in the form of a fuzzy inference system constituting a few important variables, which serves as a perceptive model for predicting the severity of sliver defects in cold rolled steel. Validation of the fuzzy system is done using actual industrial trials.

1 Introduction

With rapid development of electronic appliances and the automobile industry, the need to use high end products without significantly compromising the rigidity and stiffness of the sheet has acquired considerable importance [1]. One of the most important characteristics of cold rolled metal sheets is its surface quality. Surface

I. Mohanty
Research and Development, Tata Steel Limited, Jamshedpur, Jharkhand 831007, India
e-mail: iti.mohanty@tatasteel.com

P. Dey
Department of Mechanical Engineering, Academy of Technology, Hooghly 712121, India

S. Datta (✉)
Department of Mechanical Engineering, SRM Institute of Science and Technology
(formerly known as SRM University), Kattankulathur 603203, Tamil Nadu, India
e-mail: shubhabp@srmist.edu.in

quality and substrate material cleanliness must meet strict requirements, especially when used as outer automotive panels and in beverage or food cans. While thin rolled sheets are subjected to flanging, as in manufacturing cans, the flanged edge may tear if coarse inclusions are present. High speed production process of continuous strips of steel leads to the generation of imperfections in the strip quality. This often creates problems in production contributing to poor quality of finished rolled product [2, 3]. Defects may arise from high non-metallic inclusion contents at steel making stage or may be caused during subsequent downstream processes such as casting, reheating, hot/cold rolling or drawing. The inclusions can be oxide particles, casting powder, alumina clusters, refractory, etc. [4].

Slivers are surface defects on strips of steel which appear as elongated metal flaps with some material entrapped below these flaps. Materials with high melting point get agglomerated in steel during casting and are entrapped in the initial solidification stage of casting. After rolling of the steel sheets these materials appear as defects on either surface of the steel [5]. It appears parallel to the rolling direction and is usually distributed irregularly over the strip width. Such defects generally remain undetected during immediate processing stages and ultimately show up in the finished product during final inspection. This camouflaging feature has still now kept the cause of sliver formation mostly elusive to the industrial as well as academic community and made its elimination and reduction one of the major issues in the domain of steel processing. While models have been previously proposed that can predict the heat transfer [6] and final micro-structural features [7] during hot rolling or estimate its texture during cold rolling [8, 9]. These models do not address sliver issues. One probable reason why the cause of sliver defect has not been adequately researched in academia and the measures to prevent the same could not be adopted in the industry is the complex interactions that take place between different variables in the successive phases of steel making process [10].

While the production of steel is a real complicated process that has evolved from the cauldron of the industrial revolution and has perfected itself during the last two centuries, the development of computing speed in the past two decades has brought about another revolution, where virtually any process from molecular interactions to origin of the universe can be simulated in silico—resulting in complicated phenomena being decoded through simple calculations approximating intricate reality. Several computational intelligence methods have seen applications in the steel industry, with neural network [10–12] and genetic algorithm being used successfully to design cold rolled steel sheets leading to achievement of desired mechanical properties [13]. Rough set [14, 15] and fuzzy [16] set theories are two computational intelligence methods that have also been successfully applied in the materials design domain for designing materials to achieve targeted performances. Rough set theory is known for its capability to reduce the dimensionality of datasets by selecting important features [17], whereas fuzzy set theory can describe objective categories so as to be perceivable subjectively [18]. Rough and fuzzy sets are two aspects, two different perspectives of modelling imprecision and uncertainty in real-life situations. The approximation spaces of rough set theory assign multiple memberships to a data-point, while fuzzy sets are concerned with parallel memberships. Exploiting their

reciprocity and complementarity, researchers have tried to integrate rough and fuzzy set theories into a single mathematical theory, often called rough-fuzzy models [19].

In the present work rough set has been used to select the important variables from the compositional and process parameters of different stages of steel sheet making process to form significant if-then-else rules, while fuzzy set theory assigns fuzzy membership functions to each of the selected variables, thus evolving a fuzzy inference system (FIS) [20, 21]. The FIS can be implemented at the production shop to monitor and warn the quality control department at every steel making and processing stage regarding pitfalls that might lead to sliver defects. The FIS will help the people working at the production site to better understand or perceive the control process due to its linguistic mode of expressing a rule, while the selected features and rules derived using rough set help to keep each rule short and limit the number of rules to a handful few. As a final step of appraisal of the results obtained, some additional data from the steel industry has been used to validate the model, where the real nature of sliver defects could be predicted by the rough-fuzzy rules developed in this paper with sufficient accuracy.

2 Dataset

After collection and assimilation, the data is cleaned to remove incomplete and incoherent values. The final version of the cleaned dataset contains 245 observations, each representing different sheets of steel with varying compositions processed under different ambient conditions, and exhibiting different levels of severity of the sliver defect. There are 45 input variables and a single output variable. The name, abbreviation, mean and standard deviation of each variable is given in Table 1.

3 Methodology

3.1 p-value

A *p*-value is a statistical measure that provides the quantity of evidence present in the dataset enabling the rejection of the most common explanation for the dataset. It can be considered to be the probability of obtaining a result at least as extreme as the one observed, given that the null hypothesis is true. The null hypothesis is considered to be the most plausible scenario that can be used to explain a set of data. The null hypothesis is always assumed to be true unless proven otherwise. An alternative hypothesis predicts the opposite of the null hypothesis and is said to be true if the null hypothesis is proven to be false [22].

A *p*-value helps in determining the significance of the result. All hypothesis tests use a *p*-value to calculate the strength of the evidence (what the data tells about

Table 1 Basic statistics of the data for sliver defect analyses

Sl. no.	Attribute name (unit)	Abbreviation	Average	Std dev
Input variables				
1	Roughing mill exit temperature (°C)	RMET	1086	16
2	Slab retention time (min)	Ret time	194	91
3	Slab dropout temperature (°C)	SDOT	1209	23
4	Hot rolled thickness (mm)	HR tk	3.8	0.8
5	Re-heating furnace oxygen content (wt.%)	RHO_2	2.2	1.03
6	Purging duration (min)	Purg dur	30	6.48
7	Casting duration (min)	Cast dur	53	7.79
8	Argon flow (Nm^3/min)	Ar flow	6.7	2.21
9	Argon back pressure (kg/cm^2)	Ar BP	0.15	0.11
10	Heat no. in Tundish	Heat TD	5	4.43
11	Tundish superheat (°C)	TD SH	28.4	6.74
12	Maximum mould level fluctuation (%)	Mlf max	75	3.26
13	Average mould level fluctuation (%)	Mlf avg	69.7	0.45
14	Minimum stopper position (%)	Stp min	63.9	9.68
15	Maximum stopper position (%)	Stp max	69.9	8.72
16	Average stopper position (%)	Stp avg	67.2	8.48
17	Minimum casting speed (mm/min)	Spd min	1150	238.2
18	Average casting speed (mm/min)	Spd avg	1226	182.15
19	Minimum tundish weight (t)	TD wt Min	26	3.37
20	Average tundish weight (t)	TD wt Avg	27	2.02
21	Aluminium addition pattern	Al pattern Add	1.4	0.56
22	Al_2O_3 (wt.%)	Al_2O_3	9.84	13.3
23	CaO (wt.%)	CaO	49.21	6.43
24	Iron (wt.%)	Fe	15.83	7.07
25	MgO (wt.%)	MgO	2.01	2.069
26	MnO (wt.%)	MnO	1.15	1.58
27	P_2O_5 (wt.%)	P_2O_5	2.31	1.43
28	Sulphur (slag) (wt.%)	S slg	0.056	0.07
29	SiO_2 (wt.%)	SiO_2	11.64	4.3
30	Carbon (wt.%)	C	0.02	0.02
31	Manganese (wt.%)	Mn	0.18	0.13
32	Sulphur (liquid steel) (wt.%)	S Liq	0.007	0.002
33	Phosphorus (wt.%)	P	0.015	0.012
34	Silicon (wt.%)	Si	0.006	0.008
35	Aluminium (wt.%)	Al	0.039	0.008
36	AlS (wt.%)	AlS	0.037	0.009

Table 1 (continued)

Sl. no.	Attribute name (unit)	Abbreviation	Average	Std dev
Input variables				
37	Chromium (wt.%)	Cr	0.026	0.004
38	Molybdenum (wt.%)	Mo	0.001	0.0006
39	Nickel (wt.%)	Ni	0.011	0.001
40	Vanadium (wt.%)	V	0.001	0.0003
41	Titanium (wt.%)	Ti	0.024	0.02
42	Niobium (wt.%)	Nb	0.002	0.006
43	Cupper (wt.%)	Cu	0.16	2.38
44	Boron (wt.%)	B	0.001	0.0003
45	Nitrogen (wt.%)	N	17.9	15.54
Output variable				
46	Sliver severity	Severity	3.44	1.01

the population). p-value helps to determine whether the hypotheses are correct or not. It is directly related to the significance level, which is an important component in determining whether the data obtained is statistically significant or not. Correct interpretation of the p-value is very much important. The p-value is a number which lies between 0 and 1. A small p-value (≤ 0.05) implies strong evidence against the null hypothesis forces to reject the null hypothesis. A large p-value (>0.05) implies weak evidence against the null hypothesis, forces not to reject the null hypothesis. p-values which are very close to the cut off (0.05) could go either way.

3.2 Rough Set Model

Rough Set Theory (RST) was proposed by Pawlak [23]. The paradigm is concerned with classification and analysis of uncertain, imprecise, or incomplete information and extraction of knowledge from real-life datasets bearing these natural characteristics. It is considered as one of the most robust approaches in data analytics.

The basic tenets underlying RST consist of lower and upper approximation of concepts in a set of data points, which define the domain of interest.

A dataset (or information system) $\mathcal{I}$ essentially consists of a set (or universe) U containing several observations (or objects), each observation here representing a definite sheet of cold rolled steel sheet. Every object is characterised by definite values for each of the conditional attributes in the set A, depending on which the value of a decision attribute d is determined. Using the formalism of Boolean logic, a typical RST algorithm searches for a minimal set of attributes based on which the decision attribute can be evaluated with the same accuracy as the whole set of attributes A. This essential set of attributes R is called the reduct.
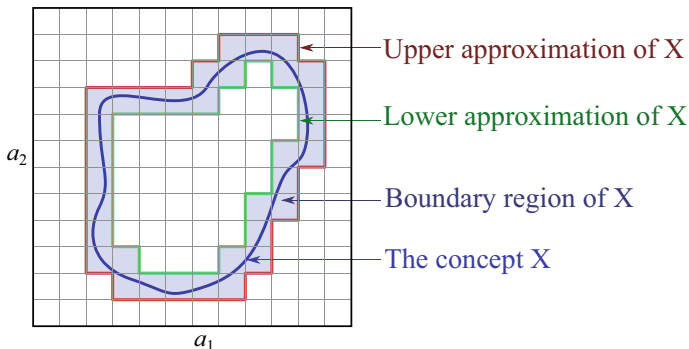
Fig. 1 Different rough set notions

When defining a concept (or classifying items as belonging to a definite class label) with the help of incomplete information, RST brings in the notions of lower approximation and upper approximation, which—unlike the actual concept—can be definitely known. A lower approximation typically contains objects that *definitely* belong to a concept, while an upper approximation contains objects that are *possibly* part of concept. The concept X is bounded by the lower and upper approximations—both of which are two different rough sets. Formally,

$$X_{\text{lower}} : \quad \underline{R}(X) = \bigcup \{Y \in U/R \mid Y \cap X \neq \emptyset\}$$

$$X_{\text{upper}} : \quad \overline{R}(X) = \bigcup \{Y \in U/R \mid Y \subseteq X\}$$

Between these two regions lie the boundary region $BN_R(X)$. These regions can be graphically represented in two dimensions (corresponding to two attributes a_1 and a_2 in R) in Fig. 1.

Rough set (RS) is an excellent tool to reduce the dimensionality of data by eliminating redundant information—both in terms of number of observations as well as number of variables (or attributes). The power of RS is illustrated by its ability to form a mechanism for classification (also called decision system) where the outcome (or decision attribute) of an observation can be predicted from its input variables (or conditional attributes) with the help of if-then rules utilising the minimum number of attributes.

The objective of the RS classifier is to frame rules of the form $\phi \Rightarrow \psi$ where ϕ is the antecedent and ψ is its consequent. A typical rule takes the form:

$$\text{If } \underbrace{a_1 = c_1 \text{ and } a_2 = c_2 \text{ and } \dots a_p = c_p}_{\phi} \text{ then } \underbrace{d = f}_{\psi}, \qquad (1)$$

where $a_1, a_2, \dots, a_p \in R$ are p different conditional attributes taking up values $c_1, c_2, \dots, c_p$, and the decision attribute d is assigned the categoric value f. A rule

remains simple and intelligible so far as the number of descriptors p in the rule is limited to a handful few. This is warranted when the size of the reduct is small, as $p \leq |R|$

For a rule to be reliable, it has to be based on a good number of observations. However, each observation is different from the other in terms of the values of its several attributes—whether conditional or decision.

To get rules of a general nature, it is necessary to partition the range of each variable into a number of sub-ranges and form rules based not on a single value of any attribute, but on a range of values of a variable (say a_i) denoted by the category c_i. This process of partitioning continuous variables into discrete categories is termed discretization [15]. Discretization is regarded as one of the crucial pre-processing steps in the synthesis of decision rules from data with real value attributes. The problem of discretization of the conditional attributes is $\mathcal{NP}$-hard and great effort has been expended in finding effective heuristics to solve this problem. Different methods from various related fields such as statistics, information theory, artificial intelligence, pattern recognition and formal logic have been proposed to tackle the discretization task [24]. In the present paper discretization of the conditional attributes has been done simultaneously with the task of finding the reduct for the dataset using the dynamic discreduction algorithm [25]. The algorithm relies on multiple samples of the data, where each sample votes for a candidate discreduct (a possible value of cut for discretization on a particular attribute). The votes cast on each attribute are added up, and the share of cuts on each attribute is decided through the principle of proportional representation. The highest voted candidates form the discretization boundaries on each attribute, and attributes getting one or more cut are included in the reduct. Thus the interdependence of discretization and selection of attributes is given due respect through proportional representation, and the experimental errors or noise in the data is filtered out through sample votes. Discretization of the conditional attribute, however, is usually carried out a priori on the basis of a simple thumb rule: the number of objects in each interval should be more or less equal. This allows each rule to collect sufficient objects to stand in its support. The decision attribute in the present investigation is categorical from its onset, but with some categories containing very few objects. A few categories are thus merged to form broader classes with the objective of bringing parity in their respective heights (i.e. number of objects in each class).

There are two important aspects to be considered while selecting a rule. From one consideration, rule built on the premise of many observations, may be valid even when contradicted by some observations. Formally some objects in $\mathcal{I}$ may match the rule's antecedent, but not satisfy the rule's consequent. Hence the probability with which the conclusion can be reached, given the condition becomes important. The accuracy of a rule gives a measure of how trustworthy the rule is in drawing a particular conclusion on the basis of its pertaining evidence [26]. From a different consideration, a rule may only portray a partial picture of the information inherent in. A rule is accepted to be strong enough if it has a good proportion of objects in its support. The coverage of a rule measures the share of the premise it uses for drawing the concerned conclusion [27].

The accuracy and coverage of a rule is computed as

$$\text{Accuracy} = \frac{\left| \, ||\phi|| \cap ||\psi|| \, \right|}{\left| \, ||\phi|| \, \right|} \tag{2}$$

$$\text{Coverage} = \frac{\left| \, ||\phi|| \cap ||\psi|| \, \right|}{\left| \, ||\psi|| \, \right|} \tag{3}$$

where $||\phi||$ and $||\phi||$ are the objects in $\mathcal{I}$ matching the antecedent and consequent of a rule, and $\left| \, \cdot \, \right|$ is the cardinality of any set. It is desirable for a rule to be accurate as well as to have a high degree of coverage.

3.3 Fuzzy Inference System (FIS)

Fuzzy logic is closer in spirit to human thinking and natural language than the conventional Aristotelian logical system. It is basically a many-valued logic based on intuitive reasoning. The fuzzy philosophy welcomes tolerance and approximation in place of precision and accuracy, thus allowing the notion of partial truth to reign with truth values lying anywhere between completely false (0) and completely true (1). In this way uncertainty in data and imprecision in knowledge is mathematically represented through a formalism [28]. Fuzzy set theory was first introduced by Lotfi A. Zadeh in the 1960s as a means to model the uncertainty of natural language. Later the application area of fuzzy logic expanded over a wide spectrum encompassing consumer products, electronic instruments to automobile, traffic monitoring and different control systems.

Fuzzy logic is extremely suitable for modelling systems where the imprecise knowledge is expressed in linguistic if-then rules involving categorical attributes rather than by exact (real-valued) mathematical expressions. It is also applicable to situations involving highly complex processes where behaviours are not well understood. In particular, problems where any feasible solution is acceptable, and precision beyond a certain level adds no extra merit, are efficiently handled by fuzzy logic. Such a situation is quite prevalent in complicated materials systems, where exact solutions are often intractable and practical situations set the limit to which a precise measure may be implemented. However, efforts made in understanding different phenomena in the field of materials modelling using FIS are still far from commendable. The surface defects of steel depend on a large number of factors, many of which are interdependent owing to the inherent dynamism of the processes leading to its formation. Furthermore, most of these factors are laden with uncertainties and ambiguities. These are the prime reasons why it is really difficult to predict the severity of the surface defects formed in steel during processing. A rule-based fuzzy modelling approach relating the possible causes of sliver formation to the severity of the defect may be considered appropriate in this situation. It is anticipated that such

an approach would reveal some of the major underlying phenomena in the genesis of sliver defects in the production of hot or cold rolled steel.

Fuzzy Inference System (FIS) is a way of mapping an input space to an output space using a collection of fuzzy membership functions and rules for reasoning about data. An FIS consists of three components, namely, the fuzzifier, the inference engine with a fuzzy rule base and the defuzzifier. The purpose of fuzzification is to map the inputs to fuzzy values between 0 and 1 using a set of input membership functions. The fuzzy rules are employed to develop the fuzzy outputs from the fuzzified inputs. The outputs are then combined to obtain one fuzzy output distribution in the aggregator, and eventually arrive at a single crisp output through the process of defuzzification [29].

For the problem under study, the sliver defect does not emanate all of a sudden, nor is it the effect of one isolated reason or factor. It is the outcome of many related processes like steel making, furnace reheating, hot rolling and cold rolling, interacting in different proportions, with only a few out of the many possible parametric levels producing a certain degree of severity of sliver defect. For any particular attribute, there is no threshold value based on which a decision on the condition of the steel sheet—whether defective or non-defective—can be taken. Problems of this kind can be modelled using fuzzy inference systems more closely.

Two most common methods to develop an FIS are the Sugeno and Mamdani methods. The basic difference between the two lies in the way of deriving crisp outputs from fuzzy inputs. The Mamdani type FIS is widely accepted for decision support applications as its rule base is easy to interpret and intuitive in nature [30, 31]. In the present study the Mamdani method has been used to develop the FIS model owing to its efficiency in capturing expert knowledge in an intuitive and human-like manner. A Mamdani FIS consists of five layers: fuzzy layer, product layer, implication layer, aggregation layer and defuzzification layer. In the first layer the crisp input values are converted to the fuzzy values by the input MFs to determine the degree to which these inputs belong to each of the appropriate fuzzy sets. In this case Gaussian MFs for the inputs are used. In the second layer firing strength of each rule is computed. The strength of each rule is determined by evaluating the membership expressions in the antecedent of the rule. This is accomplished by the input MFs in the layer 1. If a given fuzzy rule has more than one antecedents, the fuzzy operator ("AND" or "OR") is used to arrive at a single number. To evaluate the disjunction of the rule antecedents the "OR" fuzzy operation is used which is given by

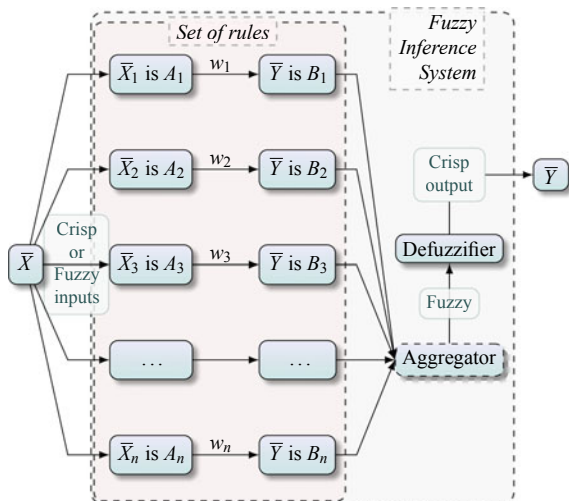$$\mu_{A \cup B}(X) = \max(\mu_A(X), \ \mu_B(X)) \tag{4}$$

To evaluate the conjunction of the rule antecedents, the "AND" fuzzy operation is used which is given by

$$\mu_{A \cap B}(X) = \min(\mu_A(X), \ \mu_B(X)) \tag{5}$$

where X is the process variable and A and B are the linguistic variables.

Fig. 2 A typical Fuzzy
Inference System (FIS) with
the set of rules and
aggregator/defuzzifier to get
a crisp output from crisp or
fuzzy inputs

After evaluating the antecedents the result is applied to the membership function of the consequent. The membership functions of the all rule consequents are combined into a single fuzzy set. A defuzzification method is applied on the aggregated fuzzy set to find a point. The most popular centroid method has been used as a defuzzification technique (Fig. 2).

4 Results and Discussion

4.1 p-value Results

Data cleaning has been performed prior to the core analysis, owing to the presence of noise in the data collected from the plant. The data was primarily subjected to p-value analysis. The most important parameter is the one with least p-value and the least important parameter is the one with highest p-value. Of the 45 variables taken, 21 are found to be important from the perspective of the criteria mentioned in Sect. 3.1. They are enlisted in Table 2 with their respective p-values. The other variables having high p-value are supposed to have insignificant influence in sliver formation. There are some interesting findings from a metallurgical perspective at this point. Formation of sulphides due to the presence of Mn and S in Table 2 can be conceived as a major source of inclusions in steel, which may be accounted as a possible cause of sliver formation. Exclusion of S through slag (S Slg), presence of MnO and addition pattern of Al (Al addition) are some of the important variables responsible for the formation of oxide during steel making. More Al is required for deoxidation, while the dissolved oxygen concentration dictates the amount of added

Table 2 Significant variables with p-value

Sl. no.	Variables	p-value
1	HR tk	0.016
2	Mlf avg	0.025
3	Ti	0.034
4	S	0.061
5	Ar flow	0.081
6	MgO	0.101
7	Mn	0.11
8	SiO_2	0.118
9	Ret time	0.172
10	RHO_2	0.175
11	P	0.175
12	RMET	0.192
13	Ni	0.197
14	B	0.243
15	N	0.302
16	FeO	0.312
17	Al pattern add	0.32
18	Nb	0.34
19	TD wt. Avg	0.379
20	TD SH	0.38
21	MnO	0.397

aluminium during tapping. As a result, an increased amount of deoxidation products like alumina is formed. Thus factual evidence suggests that sliver formation might not have any dependency on the *amount* of Al content. Rather it depends on the *addition pattern* of Al which is associated with the formation of aluminium oxide and its possible entrapment. Entrapment of refractory particles detached from the mould wall is a result of mould level fluctuation during continuous casting, which leads to entrapment of mould flux in the sub-surface region of the solidifying strand. Sliver defects in the cold rolled sheets [32] may be generated from this entrapped inclusions. But the important finding is, not the maximum fluctuation but the *average value* is the decisive factor determining the severity of sliver defects. So it can be said that higher value of average mould fluctuation impacts significantly on steel making rather sudden fluctuation. We can find the same thing in case of tundish weight, where the *average weight*, rather than the minimum weight, is significant. The oxygen content of the reheating furnace and the retention time—both suggest the possibility of formation and entrapment of oxide during rolling. The phenomena of internal oxidation, leading to iron oxide scale is very common in case of sliver, suggesting that slab defects develop during the reheating stage in the furnace. The

internal oxidation that is a predominant feature of sliver defects, suggest that the defective region might have been exposed to a high temperature for quite a long period—which is plausible in a reheating furnace. Higher temperature of discharging or longer **slab retention time** in the reheating furnace often involve the production of more scales, with more cooling water required and higher risk of damaging rolling mills (due to thermal stresses or accumulation of scale on the inner surface of the furnace). Thickness of the hot rolled product seems to be the most significant factor. However, on the other hand, casting speed and or stopper position do not seem to have any effect on the sliver defect. To sum up the most important findings in this section: the cause of sliver formation may be attributed to different factors spread across the entire production process of the production of cold rolled sheets, rather than any particular stage of the production process be held responsible for it.

4.2 Rough Set Results

The reduct is computed by using the procedures explained in detail in Sect. 3.2. The variables in the reduct achieved in the process with the number of cuts are shown in Table 3. It is seen here that Rough Set could reduce the number of important variables to 21. The set of 45 inputs could be reduced to 13 variables by the feature selection procedure using p-value as mentioned in Table 3. But it is interesting to note that these 13 variables are not a strict subset of the 21 variables selected with Rough Set alone.

Only 7 of the attributes are found to be common in both the cases, and they are **Ar flow, Mn, MgO, RET time, RH O_2, RMET** and **TD SH**. It could be easily said that

Table 3 Variables and cuts selected in the reduct from the full data with 45 attributes

Sl. no.	Variables	No. of cuts	Cut values
1	RMET	3	1043, 1077, 1094
2	Retention time	1	170
3	SDOT	3	1189, 1202, 1223
4	RH O_2	2	1.57, 1.98
5	Ar flow	1	7
6	TD SH	1	21
7	Stp max	1	66.62
8	Speed min	1	1070
9	CaO	1	50
10	MgO	1	0.58
11	Mn	1	0.38
12	AlS	1	0.08
13	Cu	1	0.029

Table 4 Variables and cuts selected in the reduct from the 21 attributes with p-value < 0.05

Sl. no.	Variables	No. of cuts	Cut values
1	RMET	3	1078, 1099, 1110
2	Ret time	1	170
3	HR tk	1	3
4	RH O_2	3	1.29, 1.98, 3.09
5	Ar flow	1	21
6	FeO	1	18.02
7	Mn	1	0.38
8	Ti	1	0.068
9	N	1	25

Table 5 The finally selected attributes in the reduct along with category boundaries

Sl. no.	Variables	No. of categories	L	M	H	VH
1	RMET	4	$(-\infty, 1078]$	$(1078, 1099]$	$(1099, 1110]$	$(1110, \infty)$
2	Retention time	2	$(-\infty, 170]$	–	$(170, \infty)$	–
3	SDOT	4	$(-\infty, 1189]$	$(1189, 1202]$	$(1202, 1223]$	$(1223, \infty)$
4	HR tk	2	$(-\infty, 3]$	–	$(3, \infty)$	–
5	RH O_2	4	$(-\infty, 1.29]$	$(1.29, 1.98]$	$(1.98, 3.09]$	$(3.09, \infty)$
6	Ar flow	2	$(-\infty, 7]$	–	$(7, \infty)$	–
7	CaO	2	$(-\infty, 50]$	–	$(50, \infty)$	–
8	MgO	2	$(-\infty, 0.58]$	–	$(0.58, \infty)$	–
9	Mn	2	$(-\infty, 0.38]$	–	$(0.38, \infty)$	–
10	S	2	$(-\infty, 0.005]$	–	$(0.005, \infty)$	–

these variables are most important as they have been identified by both the methods. Among these variables some are directing towards the oxide formation on the metal body prior or after hot rolling. To get more tapered selection of reduct, the algorithm is applied to the data consisting 21 parameters selected earlier. Here it is found that the number of variables in the reduct is reduced to 9. These 9 attributes (along with the corresponding cuts) are described in Table 4. If these 9 variables are compared with the 13 variables of Table 3 selected directly from all variables through Rough set analysis, one will see that only 5 variables are common in both the cases. Those five variables are Ar flow, Mn, RMET, RET time and RH O_2.

The above exercises of reduct selection through rough set analyses have provided some definite lead for selecting the variables related to formation of slivers in cold rolled sheets. Considering the two reduct sets given above, the statistical analysis results and the available domain knowledge, final set of 10 important variables is chosen as described in Table 5.

The value of Severity is ascertained to a rule by giving it the value suggested by the majority of objects in which the rule fires (see Eqs. 2 and 3).

Suitable linguistic labels are assigned to each interval. The accuracy and coverage (in %) are, respectively, given in square brackets at the end of each rule.

A primary list of 7 rules generated with at least 80% accuracy and 15% coverage are given below:

Rule Set 1
1. If RMET = **L**, SDOT = **L**, HR tk = **H**, RH O_2 = **M**, CaO = **L** then Severity = **M** [85.0, 15.6]
2. If RMET = **L**, SDOT = **L**, HR tk = **H**, RH O_2 = **M**, CaO = **L**, S = **H** then Severity = **M** [81.0, 15.6]
3. If RMET = **L**, Ret Time = **H**, SDOT = **H**, HR tk = **L**, RH O_2 = **M**, Ar flow = **L**, Mn = **L**, S = **H** then Severity = **M** [82.6, 17.4]
4. If RMET = **L**, Ret Time = **H**, SDOT = **H**, HR tk = **H**, RH O_2 = **M**, Ar flow = **L**, Mn = **L**, S = **L** then Severity = **M** [82.6, 17.4]
5. If RMET = **M**, Ret Time = **H**, SDOT = **VH**, HR tk = **L**, RH O_2 = **M**, CaO = **L**, MgO = **L**, Mn = **H** and S = **L** then Severity = **H** [83.3, 15.4]
6. If RMET = **H**, Ret Time = **H**, SDOT = **L**, RH O_2 = **M**, Ar flow = **H**, CaO = **L**, MgO = **H** and Mn = **H** then Severity = **M** [85.7, 16.5]
7. If RMET = **VH**, SDOT = **M**, HR tk = **L**, Ar flow = **L**, MgO = **H**, Mn = **H** and S = **H** then Severity = **H** [83.3, 15.4]

The full list of 171 rules is not provided here, as it will occupy unnecessary space without adding any important information. If the number of rules is high, the total coverage will be better which would define the system more completely. But it will be difficult to find useful and perceptible knowledge from crowded series of rules [33]. So from the knowledge extraction point of view a shorter rule set is preferred to get a clearer understanding of the system. A close study of the 7 rules listed above clearly outlines the effects of attributes in interaction and mutual combination. As a whole, the rules give a holistic idea about the role of variables in determining the severity of sliver defects in cold rolled steel sheets.

It is clear from this analysis that both steel making and the hot rolling process has substantial contribution toward the formation of sliver defect in the finishing stage of products. Elements like argon flow rate can be adjusted by changing the geometry of the nozzle geometry and casting-related conditions to have a feasible trade-off between rendering the hot flow to the surface and avoiding too much turbulence at the surface. In turn this impacts the entrapment of inclusions in the liquid steel. Inclusions coming from many sources, including argon flow, oxide inclusions generated during steel making processes are carried in with the steel entering the mould cavity and slag entrainment. If the liquid steel is not cleaned at this stage effort is made to remove it at the slab stage during scarfing. These surface inclusions lead to sliver in the final product. High sulphur content has come out to be a significant parameter in this analysis. This is because the flow direction of molten steel is along the interface of the solid and liquid phases, caused by electromagnetic stirring in the mould. This

subsequently reduces the bubbles formed at the interface of the solidified shell by inhibiting development of blowholes on the sub-surface of the slab. Blowholes are further inhibited by a relatively lower sulphur content. It can also be inferred from this analysis that RMET, SDOT and reheating furnace oxygen content RHO_2 are the primary contributors to sliver formation. Improved control of the furnace atmosphere enables to have a more stable oxygen content inside the furnace. However, most metals undergo some kind of oxidation at the surface, resulting in loss of material leading to poor surface finish. By improving the reheating process sliver defect can be amicably reduced.

After identifying the most influencing parameters, a fuzzy inference system (FIS) is developed based on the selected rules to predict the defect severity.

4.3 Development of FIS Model

Based on the set of rules developed using rough set theory three fuzzy rule-based models [34, 35] are developed with 7 rules, 34 rules and 171 rules, respectively. The rules which were derived using rough set algorithm are the inputs to build a fuzzy inference engine using membership functions. The membership functions are the one which defines how each input space is mapped into a membership value (or degree of membership) between 0 and 1. It is mainly derived from the cut values which are given in Table 5. Based on that variables are mapped into different groups like low (L), medium (M), high (H) and very high (VH). The generalised Gaussian membership function is used here. This function depends on two parameters called mean and variance, which is calculated by putting the cut values and minimum and maximum value of the attributes in the equation

$$y = e^{-\frac{(x-c)^2}{2\sigma^2}} \tag{6}$$

$$\Rightarrow \quad \sigma = \sqrt{\frac{-(x-c)^2}{2 \ln y}} \tag{7}$$

where (c, σ) is a set of distinct centres and width vectors of a membership curve. The ten inputs are denoted as

$$\mu_A^{(Ar\ flow)} \qquad \text{where } A \in \{\mathbf{L}, \mathbf{H}\} \tag{8}$$

$$\mu_B^{(CaO)} \qquad \text{where } B \in \{\mathbf{L}, \mathbf{H}\} \tag{9}$$

$$\mu_C^{(MgO)} \qquad \text{where } C \in \{\mathbf{L}, \mathbf{H}\} \tag{10}$$

$$\mu_D^{(Mn)} \qquad \text{where } D \in \{\mathbf{L}, \mathbf{H}\} \tag{11}$$

$$\mu_E^{(S)} \qquad \text{where } E \in \{\mathbf{L}, \mathbf{H}\} \tag{12}$$

$$\mu_F^{(\text{HR tk})} \qquad\qquad \text{where } F \in \{\mathbf{L}, \mathbf{H}\} \tag{13}$$

$$\mu_G^{(\text{RH O}_2)} \qquad\qquad \text{where } G \in \{\mathbf{L}, \mathbf{M}, \mathbf{H}\} \tag{14}$$

$$\mu_H^{(\text{SDOT})} \qquad\qquad \text{where } H \in \{\mathbf{L}, \mathbf{M}, \mathbf{H}, \mathbf{VH}\} \tag{15}$$

$$\mu_I^{(\text{Ret Time})} \qquad\qquad \text{where } I \in \{\mathbf{L}, \mathbf{H}\} \tag{16}$$

$$\mu_J^{(\text{RMET})} \qquad\qquad \text{where } J \in \{\mathbf{L}, \mathbf{M}, \mathbf{H}, \mathbf{VH}\} \tag{17}$$

Figure 3 shows the membership functions of some of the input variables as generated using the cuts found in the RS analyses in the above method. Figure 4 shows the same for the output variables. Every rule derived from rough set theory represents a fuzzy relation between the inputs and the defect severity. The inference rules con-
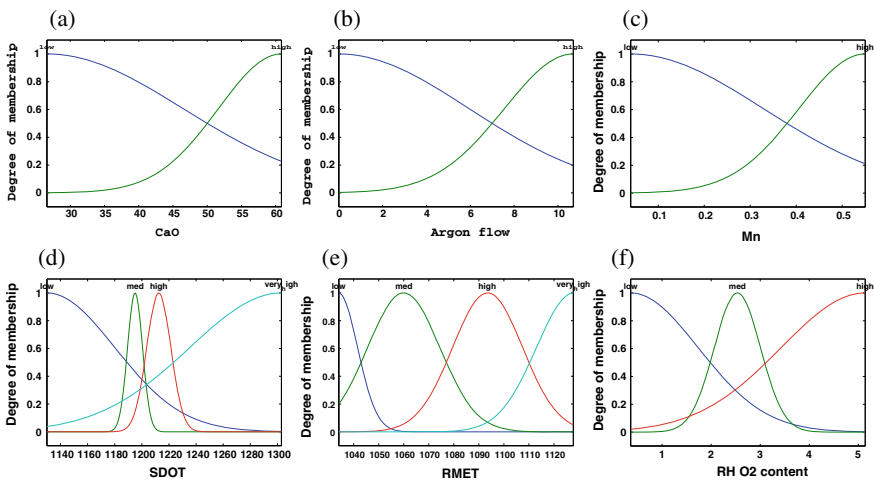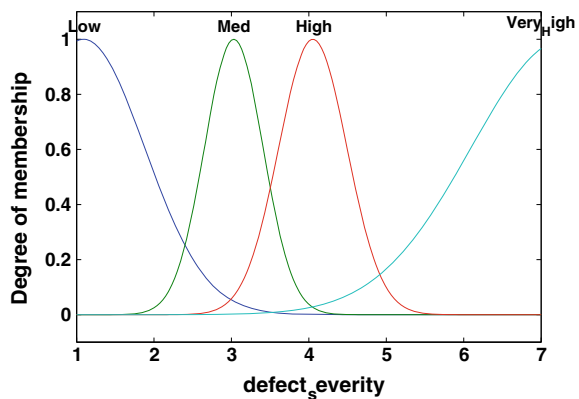

Fig. 3 Membership functions for **a** CaO, **b** Ar flow, **c** Mn, **d** SDOT, **e** RMET, **f** RH O$_2$

Fig. 4 Membership function for defect severity

Table 6 Tabular form of Rule Set 1

| Rule #	Ar flow	CaO	MgO	Mn	S	HR tk	RH O$_2$	SDOT	Ret time	RMET	Severity
1		L				H	M	L		L	M
2		L			H	H	M	L		L	M
3	L			L	H	L	M	H	H	L	M
4	L			L	L	H	M	H	H	L	M
5		L	L	H	L	L	M	VH	H	M	H
6	H	L	H	H			M	L	H	H	M
7	L		H	H	H	L		M		VH	H

necting these inputs to output consists of four membership functions (Low, Medium, High and Very High). Fuzzy rules are always written in the form of "If (input1 is membership function1) and/or (input2 is membership function2) and … then (output is output membership function)", which is given in the tabular form in Table 6. The results of various rules are summed together to generate a set of "fuzzy outputs". By using the conjunction operator (AND) in the antecedents of the rules, the rule firing strength is calculated. The antecedent in each rule describes to what degree the rule applies, while the conclusion assigns a fuzzy function to output variable. After combining the consequence of the rules by combining the rule strength and the output membership function, output distribution is achieved.

At last it is necessary to come up with a single crisp output from the FIS. This crisp number is obtained in a process known as defuzzification. It converts the fuzzy output of the inference engine to crisp using membership functions analogous to the ones used by the fuzzifier. Here centroid method has been used as defuzzification where the crisp value of the output variable is computed by finding the variable value of the centre of gravity of the membership function for the fuzzy value. It gives the centre of the area under the curve.
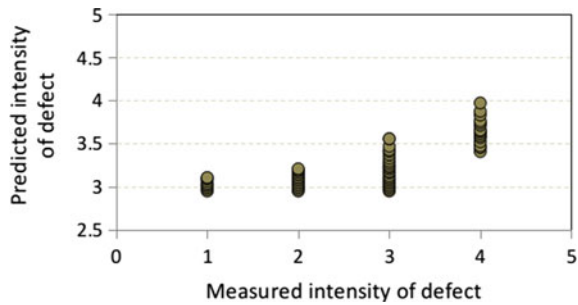
4.4 FIS Prediction

The rules developed by the RS analyses and the membership function developed are used for creating a fuzzy inference systems (FIS) for predicting the severity of the slivers, where the ten attributes finally selected to be important are used as input parameters and severity of defect is considered as output. For the three sets of rules, as described earlier, three separate FISs are developed. The surface plots showing the role of the variables, generated using the FIS from 171 rules, are shown in Fig. 5. The surface plots generated in some cases are found to be quite complicated. But the general trends of the plots show that other than the rolling parameters related to oxide formation in the surface CaO and S content play important role. These studies
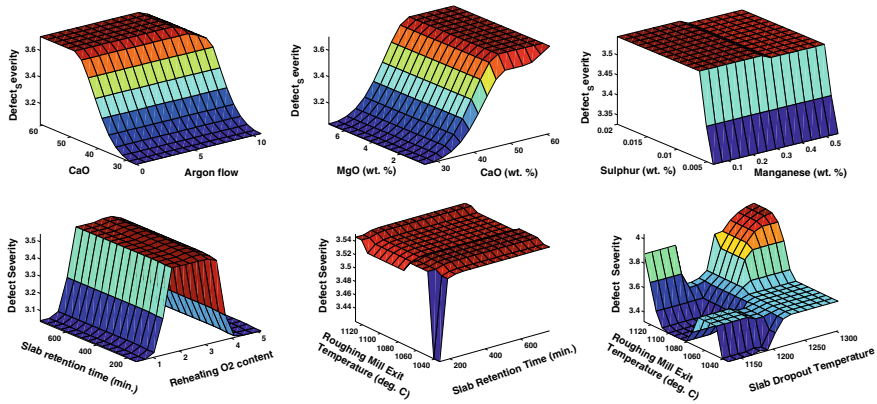
Fig. 5 Surface plot of Defect Severity against **a** CaO and Ar flow, **b** CaO and MgO **c** S and Mn, **d** Ret time and RH O$_2$, **e** RMET and Ret time, **f** RMET and SDOT
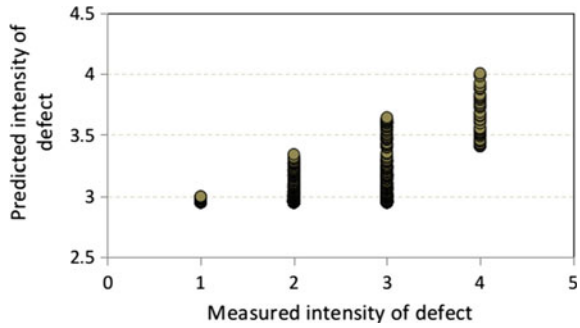
**Fig. 6** Plot of predicted defect severity against actual defect severity in case of 171 rules

using the FIS point towards producing clean steel along with control in the hot rolling parameters, in the direction of reducing the formation of oxide layer.

The predictions of the three Fuzzy Inference Systems are shown in Figs. 6, 7 and 8. There is not much to compare among the predictions, as in all the cases the shortcomings are similar. It seems that the predictions for higher severity are consistent in all the cases, whereas it is quite poor in case of lower severity particularly in case of severity level 1. This observation seems to be expected as in case of lower severity of defect it becomes more difficult to assess the cause of the defect. In this situation the relations among the variables are obscure and difficult to express through the rules. In case of the rules generated from RS analyses, it could be found that the number of rules for low severity condition is quite low. Thus the FISs generated could not be used successfully for detecting the low severity of defects.

Fig. 7 Plot of predicted defect severity against actual defect severity in case of 34 rules

Fig. 8 Plot of predicted defect severity against actual defect severity in case of 7 rules
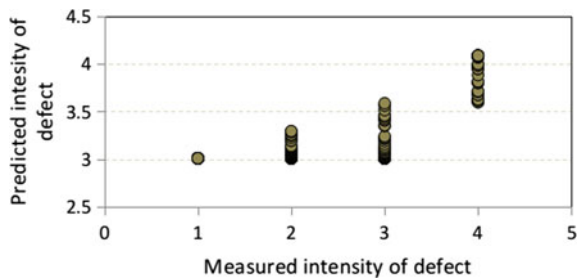
5 Experimental Trials for Validation

For the experimental verification of the above model for predicting severity of the defects, five samples with different composition and process history are collected. The samples were cleaned and studies under scanning electron microscopy (SEM) were carried out along with EDX analyses. Representative micrographs and chemical analyses of the slivers are reported for all five cases. Table 7 shows the compositional and process history of the samples along with their severity. The level (low, medium, etc.) of the attributes according to the RS analyses are described in the parenthesis. Table 8 shows the EDX analyses result describing the chemical composition of the sliver defects. The chemical compositions clearly indicate that the slivers are basically oxides of different elements. Existence of Ca in all cases clearly indicates that CaO from slag plays an important role in slag formation, which was identified by the data mining efforts also. In Table 7, it is seen that Cao in slag is high for all cases. Here the major difference between the three sliver compositions is that the content of Al, Si, Mn and Fe varied from case to case. Though SiO_2 was not identified as important parameter, but it seems to have a presence in two of the three cases. Similarly Al_2O_3 is another oxide, which has some role to play.

Thus the experimental result also indicates that sliver formation is not the effect of any particular attribute or any particular process. It is a cumulative effect of different oxides including the iron oxides, which may come during the hot rolling process. These oxides are results of oxygen diffusion and precipitation of FeO under high

Table 7 Compositional and process history of samples

Sample #	Ar flow	CaO	MgO	Mn	S	HR tk	RH O₂	SDOT	Ret time	RMET	Severity
1	5.9	68.21	2.83	0.18	0.006	2.8	3.81	1141	381	1040	2
	(L)	(H)	(L)	(L)	(H)	(L)	(H)	(L)	(H)	(L)	(M)
2	7.2	58.26	5.01	0.18	0.007	2.6	1.49	1202	142	1060	3
	(H)	(H)	(H)	(L)	(H)	(L)	(L)	(M)	(L)	(M)	(H)
3	5.3	50.35	2.45	0.17	0.006	3.0	5.80	1209	159	1096	2
	(L)	(H)	(H)	(L)	(H)	(L)	(H)	(H)	(L)	(H)	(M)
4	4.6	58.34	4.24	0.14	0.005	4.0	4.92	1142	201	1106	2
	(L)	(H)	(H)	(L)	(L)	(H)	(H)	(L)	(H)	(H)	(M)
5	5.1	32.11	4.31	0.50	0.007	3.8	5.11	1153	403	1055	3
	(L)	(L)	(H)	(L)	(H)	(H)	(H)	(L)	(H)	(M)	(H)

Table 8 EDX analyses result of the samples

| Sample #	O	Na	Al	Si	K	Ca	Mn	Fe
1	49.78	5.92	5.45	15.93		15.49	1.63	5.80
2	39.26	6.11	13.79	0.71	1.32	12.63	20.26	5.92
3	28.08	1.71	1.22	10.11	0.42	18.53	0.61	39.32
4	24.77	2.59	2.66	5.31	0.35	5.21	1.99	57.12
5	29.21	1.21	10.72	3.19	0.44	4.25	15.55	35.43

temperature. These precipitates are in solid state, composed mainly of FeO, often accompanied with MnO and SiO$_2$.

In cases of samples 3, 4 and 5, the amount of Fe in the slivers are quite high, indicating the presence of iron oxides. Figure 9a–e shows the physical appearances of the slivers under SEM. Once internal oxidation is detected that could help in separating a few of the probable sources of sliver formation, and in identifying the plausible origin of sliver. Internal oxides are known to form only at elevated temperatures where the concentration of oxygen is high enough, and exposed for quite a prolonged period of time. Such temperatures usually occur during the process of slab casting, slab reheating, before hot rolling and also in the early stages of hot rolling or roughing.

6 Conclusion

Two state-of-the-art computational intelligence tools, viz., Rough and Fuzzy Set Theories, were employed to determine the cause of sliver defects in the formation of hot or cold rolled steel sheets. While Rough Set Theory (RST) helps to determine the important variables responsible for sliver formation in addition to statistical evidence, a fuzzy information system (FIS) defined on rules derived from RST is used to predict

Fig. 9 Physical appearances of five sliver samples under SEM
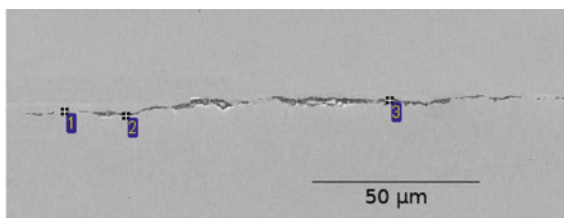
(a)
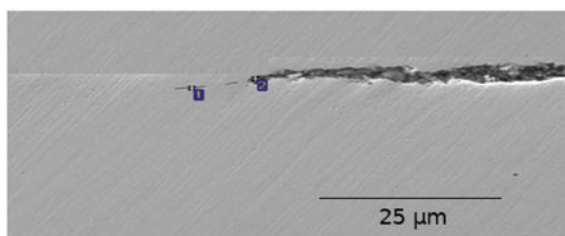
(b)

(c)

(d)

(e)

the severity of the defects. Validation of the data-driven model is done through selected experimental trials in the industry. The study may be concluded with the following summary observations.
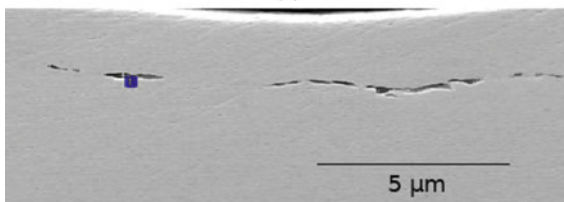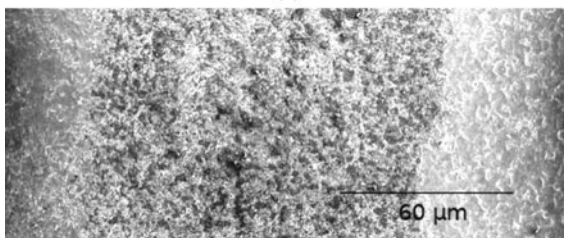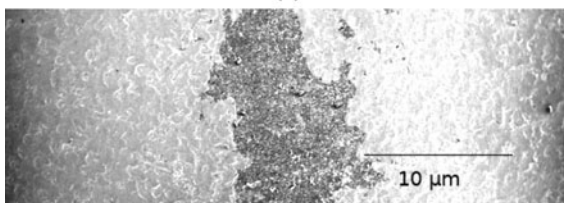
1. Despite rigorous study, the root cause of sliver formation does not seem easy to figure out. Slivers on the surface of cold rolled steel strips can either be from foreign inclusions or due iron oxides formed on it.
2. The important variables belong to all the phases of the steel processing, which leads to the conclusion that no particular process variable or processing stage could be identified as the standalone cauldron of sliver defects.
3. The generated rules clearly show that both the process of steel making, as well as reheating or roughing play a pivotal role in the formation of sliver. In some cases it is not one single factor, but a myraid of different factors affecting together that give birth to sliver defects of varying severity.
4. Among the process variables, hot rolling parameters like reheating furnace oxygen content (RHO_2), slab drop out temperature ($SDOT$) and roughing mill exit temperature ($RMET$) seem to play more determining and sensitive roles in generating slivers.
5. While p-value is a classical statistical measure to judge the significance of a host parameters in predicting some target variable where their interdependence is rather linear, Rough Set reduct is capable of selecting the important parameters when the mode or pattern of dependence is not known.
6. Rules derived through Rough Set analysis gives a set of rules with varying cardinality or length, which can be used to predict the class of a target variable.
7. Fuzzy Inference System is found to be a robust tool for predicting the severity of slivers from a reasonably sized training set.
8. The Rough-Fuzzy model's limitation in predicting sliver defects of lower severity can be attributed to the overt uncertainty in the system, though it calls for more in-depth investigation.

References

1. Raja, B. V. (2006). *Status of cold rolled steel sheets industry in India*. Steel World: SAIL.
2. Bleck, W., Bode, R., & Hahn, F. J. (1990). Interstitial-free steels processing, properties, and application. In R. Pradhan (Ed.), *Metallurgy of vacuum-degassed steel products* (pp. 73–90). Warrendale, PA: TMS.
3. Tokunaga, Y., & Yamada, M. (1985). Method for the production of cold rolled steel sheet having super deep drawability. US Patent 4, 504, 326.
4. Sanam, V., Patra, P. K., Siddabathula, S., Das, R., & Usharani, V. (2009). Reduction of slivers due to nonmetallic inclusions in continuous casting. *Materials Science and Technology 2009 Conference and Exhibition*, pp. 1031–1041.
5. Gao, W. F. (2012). Formation and prevention of sliver defects on the surface of cold-rolled strip. *Advanced Materials Research, 402*, 221–226.

6. Saha, J. K., Kundu, S., Chandra, S., Sinha, S. K., Singhal, U., & Das, A. K.: Mathematical modelling of roll cooling and roll surface stress. *ISIJ International, 45*(11), 1641–1650 (2005). The Iron and Steel Institute of Japan.

7. Kundu, S., Mukhopadhyay, A., Chatterjee, S., & Chandra, S. (2004). Modelling of microstructure and heat transfer during controlled cooling of low carbon wire rod. *ISIJ International*, *44*(7), 1217–1223.

8. Kundu, S. (2009). Prediction of transformation texture under complex rolling condition. *Materials Science and Engineering: A*, *516*(1–2), 290–296.

9. Lenka, S., Kundu, S., Chandra, S., & Singh, S. (2013). Effect of recalescence on microstructure and phase transformation in high carbon steel. *Materials Science and Technology*, *29*(6), 715–725.

10. Mohanty, I., Bhattacherjee, D.: Artificial neural network and its application in steel industry. In: Computational Approaches to Materials Design: Theoretical and Practical Aspects, pp. 267–300. IGI Global (2016).

11. Mohanty, I., Bhattacharjee, D., & Datta, S. (2011). Designing cold rolled if steel sheets with optimized tensile properties using ANN and GA. *Computational Materials Science*, *50*(8), 2331–2337.

12. Mohanty, I., Datta, S., & Bhattacharjee, D. (2008). Composition-processing-property correlation of cold-rolled if steel sheets using neural network. *Materials and Manufacturing Processes*, *24*(1), 100–105.

13. Mohanty, I., Sarkar, S., Jha, B., Das, S., & Kumar, R. (2014). Online mechanical property prediction system for hot rolled if steel. *Ironmaking & Steelmaking*, *41*(8), 618–627.

14. Dey, S., Datta, S., Chattopadhyay, P., & Sil, J. (2008). Modeling the properties of trip steel using afis: A distributed approach. *Computational Materials Science*, *43*(3), 501–511.

15. Dey, S., Dey, P., Datta, S., & Sil, J. (2009). Rough set approach to predict the strength and ductility of trip steel. *Materials and Manufacturing Processes*, *24*(2), 150–154.

16. Datta, S., & Banerjee, M. (2005). Neuro-fuzzy modelling of the strength of thermomechanically processed hsla steels. *Indian Journal of Physics*, *79*, 473–483.

17. Jelonek, J., Krawiec, K., & Slowiński, R. (1995). Rough set reduction of attributes and their domains for neural networks. *Computational Intelligence*, *11*(2), 339–347.

18. Zimmermann, H. J. (2011). *Fuzzy set theory–And its applications*. Springer Science & Business Media.

19. Pal, S. K., & Mitra, P. (2004). Case generation using rough sets with fuzzy representation. *IEEE Transactions on Knowledge and Data Engineering*, *16*(3), 293–300.

20. Jang, J. S. (1993). Anfis: adaptive-network-based fuzzy inference system. *IEEE Transactions on Systems, Man, and Cybernetics*, *23*(3), 665–685.

21. Majumdar, G., Oraon, B., Laha, A., Ghosh, S., Mohanty, I., & Datta, S. (2010). Developing rules bases on the influence of welding parameters in fcaw process from ann model. *International Journal of Mechatronics and Manufacturing Systems*, *3*(1–2), 155–164.

22. Ernst, M. D., et al. (2004). Permutation methods: A basis for exact inference. *Statistical Science*, *19*(4), 676–685.

23. Pawlak, Z. (1982). Rough sets. *International Journal of Computer & Information Sciences*, *11*(5), 341–356.

24. Bazan, J. G., Nguyen, H. S., Nguyen, S. H., Synak, P., Wróblewski, J.: Rough set algorithms in classification problem. In *Rough set methods and applications* (pp. 49–88). Springer (2000).

25. Dey, P., Dey, S., Datta, S., & Sil, J. (2011). Dynamic discreduction using rough sets. *Applied Soft Computing*, *11*(5), 3887–3897.

26. Silverstein, C., Brin, S., & Motwani, R. (1998). Beyond market baskets: Generalizing association rules to dependence rules. *Data Mining and Knowledge Discovery*, *2*(1), 39–68.

27. Duntsch, I., & Gediga, G. (2000). Rough set data analysis: A road to non-invasive knowledge discovery. Methodos Publisher, London, UK.

28. Zadeh, L. A. (1965). Fuzzy sets. *Information and Control*, *8*(3), 338–353.

29. Negoiţă, C. V., & Ralescu, D. A. (1975). *Applications of fuzzy sets to systems analysis*. Springer.

30. Guney, K., & Sarikaya, N. (2009). Comparison of mamdani and sugeno fuzzy inference system models for resonant frequency calculation of rectangular microstrip antennas. *Progress In Electromagnetics Research*, *12*, 81–104.
31. Kaur, A., & Kaur, A. (2012). Comparison of mamdani-type and sugeno-type fuzzy inference systems for air conditioning system. *International Journal of Soft Computing and Engineering (IJSCE)*, *2*(2), 323–325.
32. Záhumenský, P., & Merwin, M. (2008). Evolution of artificial defects from slab to rolled products. *Journal of Materials Processing Technology*, *196*(1–3), 266–278.
33. Wu, W. Z., Mi, J. S., & Zhang, W. X. (2003). Generalized fuzzy rough sets. *Information Sciences*, *151*, 263–282.
34. Ross, T.J., et al. (2004). *Fuzzy logic with engineering applications* (vol. 2). Wiley Online Library
35. Setnes, M., Babuska, R., Kaymak, U., & van Nauta Lemke, H.R. (1998). Similarity measures in fuzzy rule base simplification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 28*(3), 376–386 (1998).

Machine Learning Studies in Materials Science

Barbara Mrzygłód, Krzysztof Regulski, and Andrzej Opaliński

Abstract Materials science research begins in laboratories with testing the properties of metals and their alloys, the properties of the material depending on the type of additives and microstructure, as well as the changes in these properties taking place under the influence of processing. The next step is modeling and simulation of processes to investigate the possibility of their control and monitoring under production conditions. Some studies relate to an ongoing process, and then the research focuses on quality control of the process, optimization, and detection of irregularities and product defects. At all stages of research, it is possible to apply the methods of machine learning to the extent chosen by the analyst or expert. These methods can be used to obtain knowledge about occurring phenomena, research planning, and designing of production processes (in accordance with the 4th paradigm of science), but they can also be data-driven models given the possibility of autonomous control of a selected aspect of production (in accordance with the idea of the 4th industrial revolution). This paper presents an overview of ML methods based on examples taken from the field of materials science discussed in terms of materials–processes–knowledge formalization.

Keywords Knowledge discovery · Data mining · Data science · Machine learning · Knowledge engineering · Ontologies · Materials science

1 Introduction: The 4th Paradigm of Science in the 4th Industrial Revolution

Science can be defined by four paradigms: 1st—empirical evidence, 2nd—scientific theory, 3rd—computational science, and 4th—data science [1]. Entering the age of Industry 4.0, forcing automatic control, and autonomous decision making in cyber-physical systems, the last two paradigms take on a new meaning. Just as there is no

B. Mrzygłód (✉) · K. Regulski · A. Opaliński
AGH University of Science and Technology, al. Mickiewicza 30, 30-059 Krakow, Poland
e-mail: mrzyglod@agh.edu.pl

numerical modeling without computer calculations, there is no autonomous decision making without machine learning. Data science defines the art of data processing and knowledge acquisition. Possibilities brought to us by machine learning seem to provide new research instruments that, like a microscope once, can provide quite new insights from many scientists. This is not always associated (although it is the most obvious and reliable way) with obtaining new data straight from the industrial processes. Often, data from already performed experimental studies, properly synthesized and combined, can provide new conclusions and results during mining the existing archive. However, this would not be possible without the structured procedure for obtaining this data provided by the data science regime, as well as further interpretation through the prism of domain knowledge.

Decision support, which is the paradigm of Industry 4.0, applies to both the production and processing of metals. Considering the concept of Industry 4.0 from the perspective of metallurgy, the characteristics of this area are important. When perceiving the metallurgical industry through the prism of processes, it should be noted that the production is complicated and consists of many stages, a very large number of process parameters determine the quality of the product, and the phenomena occurring during material processing are usually non-linear and difficult to model using deterministic (numerical) methods. Wherever metallurgical processes cannot be modeled using CAx tools, i.e., deterministic (numerical) models, or where it is too computationally expensive and time-consuming, stochastic models can be created to support decision making using data mining techniques.

The projects in the field of Industry 4.0, currently implemented in plants manufacturing metal products, relate to multi-level systems, where the integration layer of data from measurements and quality control is integrated with the layer of control and monitoring of processes and with the decision support layer. In such systems, problems are related not only to the problem of scale, i.e., BigData, but also to the integration of the data and machine learning models referring to quality measurements. Most often, process data and data on quality indicators come from different sources, with different sampling, benchmarks, and different means of identification. The challenges also lie in the creation of appropriate architectures of sensor systems, usually wireless, connected in the IoT network, integration of various data formats, creation of training sets, and after the implementation of data-driven models, implementation into the process and decision support.

Machine learning methods used in materials science cannot be implemented without an appropriate hardware and software infrastructure that ensures the efficiency and effectiveness of their operation and the security of data on which they operate. Most often they are implemented as part of IT systems operated and integrated with the infrastructure of the environment specific for a given industry sector. Depending on this context, the specific features of such systems may differ, but one universal model can be distinguished, representing the structure of such systems. It usually consists of three main layers, and its diagram is presented in Fig. 1. The highest layer in this type of system is composed of the interfaces, which allow the user to enter data into the system, manage this system, and apply the results returned by the methods used in the systems. Data can be entered into the system based on the
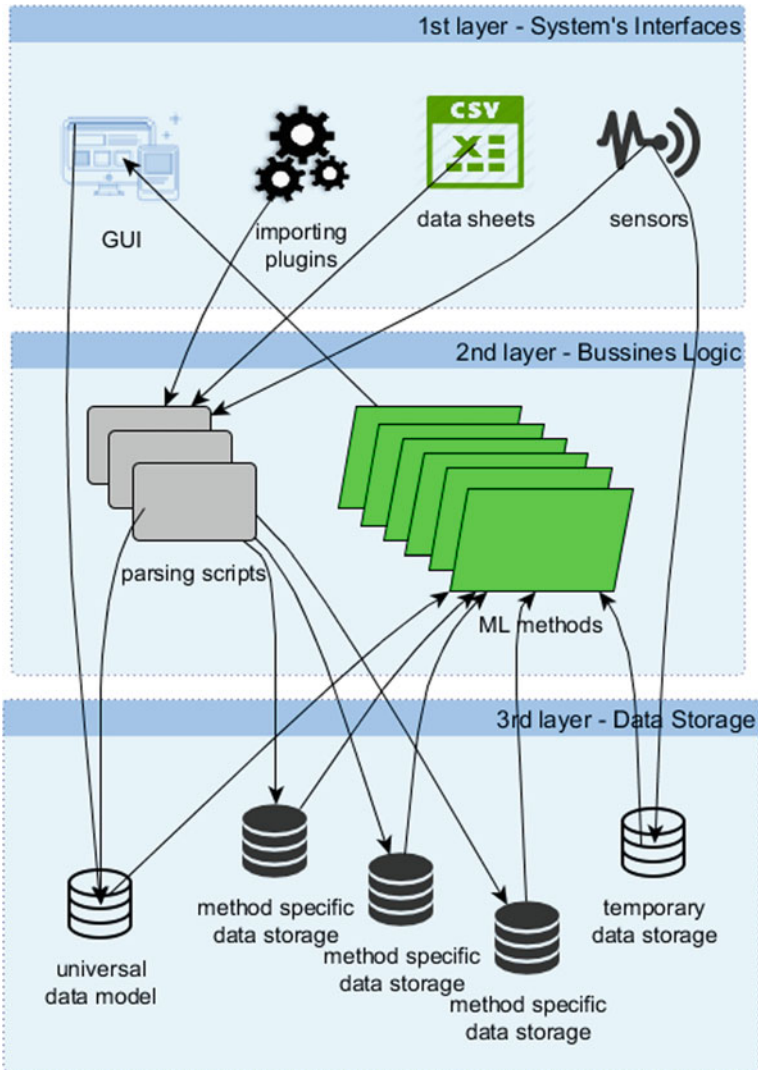
Fig. 1 Hardware and software infrastructure of machine learning systems in Industry 4.0

graphical user interface, where the system administrator or technology expert enters data based on previously prepared, dedicated sets of forms.

The second group of methods does not require direct human intervention. The first step may be to obtain data based on the provided electronic data sheets (e.g., experimental or research results). Automatic methods usually require the use of dedicated plugins (automatic software libraries that enable data acquisition from external IT systems). The third group of methods of data acquisition is based on hardware sensors, supplying directly the database layer. The methods of automatic

data acquisition usually use the functionality implemented in the second layer of the system (business logic layer), such as raw data parsers, necessary to obtain cleaned, normalized, and filtered data from the sources on which they operate. These types of mechanisms additionally ensure data consistency and completeness and monitor the sources for their availability. The business logic layer of the IT system also includes the key components of machine learning methods. Depending on the method used in the system, these may be methods of training, testing, validation of classification methods, neural networks or decision trees, or other methods, such as the support vector machine (SVM) or clustering algorithm discussed in this study. On the other hand, the components of this level use the structure and content of the third lowest layer operating in this type of information system as the basis for their functioning. It is a layer related to data and information storage models. We can find from these model that both raw data originating from, e.g., sensors or current measurements, data placed in a universal model, resulting from the operation of parsing mechanisms, and data typical of specific machine learning methods, generated for the purpose of optimal and effective data processing by these methods. The results of operation of such systems are returned to the user from the business logic layer (as the results of ML methods) to the top layer of the system (interface layer), where they are usually presented in the graphical user interface (for system users) or in the digital form (e.g., in the form of web services) for further, automatic use in more complex production systems (operating based on, e.g., the Industry 4.0 model). This type of general scheme of the system architecture can be extended to include additional components, enabling their integration with additional, specific data sources, and also to make them available in a wider information processing process.

As can be seen from the above considerations, the 4th paradigm of science in the 4th industrial revolution is primarily data acquisition, processing and storage, data mining, data modeling, and, consequently, the implementation of data-driven models. This study focuses mainly on the creation of models and machine learning and also on the subsequent formalization of acquired knowledge for later use and re-use.

2 Methods

Typically, the literature related to machine learning methods lists methods in the supervised and unsupervised division [2–4]. Supervised learning enables the prediction of qualitative (classification) or quantitative (regression) variables. This situation occurs when we have the results (answers and measurements) of a dependent variable and are able to formulate appropriate hypotheses. In materials science, the most common example of supervised learning is the prediction of material properties depending on their specificity in the form of microstructure or chemical composition. When we are looking for similar objects, signal groups, or areas of similar density, we are talking about methods of unsupervised learning—such examples can

be found in morphological analysis, in grouping materials with similar characteristics, or in recognizing the phase composition of microstructures. Examples of such studies will be discussed in this article. In particular cases, however, methods from different groups are usually used as combined or hybrid solutions. Additionally, as a separate group of applied methods of artificial intelligence, various knowledge representation formalisms that have different properties in terms of their further use and reuse will be described. Models in the form of rules (symbols) were used as a result of algorithms of rule induction, as well as fuzzy logic inference models or models in the sub-symbolic form of neural or neuro-fuzzy models ending with ontologies.

3 Materials–Processes–Knowledge Formalization

This article adopts the order of research sequences from materials (laboratory) through processes (partially controlled conditions) up to the implementation of models in the form of inference models. Each block of research has different conditions and often the analyst's point of view is different, as well as the purpose of modeling. The article summarizes many years of work on the application of machine learning methods in acquiring technological knowledge and its formalization for the needs of materials engineering and metallurgy. Table 1 summarizes the applications of these techniques, which will be expanded later in the article.

Table 1 Examples of applications of machine learning methods in material science

Materials	Processes	Knowledge formalization
Properties approximation based on: • chemical composition • processing parameters • microstructure	Process optimization by identifying the impact of parameters through: • discovery of dependencies between process parameters • study of the strength of the influence of process components • verification of the rules obtained algorithmically in the range of the variable optimum	Creation of knowledge bases for the control of processes: • induction of a set of rules • data structures for decision support systems
Identification of alloy components and classification of materials	Modeling the variability of discrete and continuous parameters: • predicting moisture content in molding sand • predicting defects of metal products • tool wear forecasting	Meta-modeling using descriptive logic: • domain ontologies • integrating heterogeneous data • re-use of knowledge components

In order to carry out the research in the area of metallurgy presented in Table 1, a number of techniques in the field of data mining, machine learning, and knowledge formalization were used. The first stage of the proposed methodology of acquiring and formalizing knowledge is obtaining data in a form that enables its further processing. The data in the presented studies came from different sources, i.e., (1) experimental data on materials; (2) data from the metal production and shaping process; and (3) data from text documents, most often PDF files containing scientific articles from the discussed field. Data from sources (1) and (2) are characterized by a similar specificity: these are usually numerical data stored in many spreadsheets and text files—their processing requires that they be integrated and combined into one database. This operation includes data transformation, cleaning, and integration based on expert knowledge. This process can be called data acquisition and integration. Taken together, the stages of work related to the creation of models and knowledge extraction make up the stages of creating an integrated, semantic knowledge base intended to support the decision-making process. Algorithmic knowledge acquisition consists of creating models based on data (data-driven models). These models are used to reflect the dependence of parameters and quantify their mutual influence. The decision (dependent) variable is chosen by the experts and described on the basis of the variability of this feature depending on the changes taking place in the other parameters. Stochastic relationships can be reflected in the form of regression models, in the form of 'IF-THEN' rules, or in a sub-symbolic form as in the case of artificial neural networks. In the conducted research, in particular, the following algorithms were used:

- induction of decision trees: CART, CHAID, random forest, and boosted trees,
- cluster analysis (k-means),
- rough sets theory,
- machine learning: SVM and neural networks.

Obtained models can be used for inference in decision support systems. They are models of inference in terms of knowledge management systems. They differ in the structure, often also in precision, but they complement each other, providing the end user with various usability. The reader will find a discussion of their functionality and the benefits of using particular methods in the publications mentioned above. The framework that completes research related to knowledge extraction from data is the formalization of this knowledge.

4   Materials

This section devoted to materials will give examples from research on the approximation of properties based on chemical composition, processing parameters, or microstructure and also from research related to the identification of alloy components and material classification. The following tools were used: (1) statistical tools, regression, (2) piecewise regression models, (3) MARSplines, (4) decision trees, (5)

random forest, (6) boosted trees, (7) artificial neural networks, (8) support vector machine (SVM), (9) rough sets theory, (10) clustering, and finally (11) neuro-fuzzy models [5, 6].

The problem most often encountered from the materials engineering perspective is predicting the future properties of the finished product to control and optimize the process. Product quality can be determined either directly by estimating the mechanical properties or indirectly by estimating the chemical composition and microstructure. The idea of Industry 4.0 proposes to monitor the technological process by comparing the current operations with the expected result, calculated in parallel from the virtual model to maintain the expected quality level. In other words, it is the constant prediction of the results of ongoing operations. In this respect, the creation of predictive models is based, first, on the search for dependencies between process parameters and then on the use of an algorithm which, by analyzing these dependencies, will be able to predict the unknown value of the parameters essential for the process efficiency (dependent variable). Research on materials is often based on the study of mechanical properties and on an attempt to predict these properties knowing the chemical composition, microstructure, or processing parameters of a given material.

Acquisition of knowledge about changes in the microstructure of silumins after the introduction of alloying elements (Mo, Cr, W, and V) could be an example of the application of machine learning techniques in the material science. The procedure for the analysis of small experimental data sets for multi-stage, multi-variate, and multi-variable models was developed during the research. The development of the voting procedure using the results of experimental studies enabled the creation of models of decision trees, the quality of which was improved with discretization. The procedure involved a combination of supervised learning with supervised learning: clustering and decision trees. Cluster analysis was used to level the multi-dimensionality of the output vector (solving the problem of many dependent variables—multi-variable). The use of discretization of both dependent variables and predictors has proved to be effective. This resulted in a multi-step classification procedure using discretization and cluster analysis. The research also showed that at each stage of the analyst's decision-making process, the conducted procedure requires a continuous search of available tools and inference on a more sophisticated level than traditional machine learning techniques. What is important is the balance between ease of use and efficiency in obtaining knowledge from small collections. DTs are especially useful with highly nonlinear problems and small datasets, and among the various tools, trees provide the simplest to interpret knowledge. The efficiency of trees can be increased by prior discretization of the variables. The use of cluster analysis made it possible to find patterns on a more general level than individual variables, thus confirming the thesis that the appropriate granularity of information allows finding patterns even in small data sets. Research results made it possible to create a knowledge base that enables decision support, improvement of quality, and shortening the time of materials testing [5]. Studies have shown that predicting properties from the chemical composition can be significantly improved by using cluster analysis to segment the

Table 2 Quality of alloy property prediction in testing the material properties

	MSE	RMSE	MAE	R	R^2
Quality of alloy property prediction based on chemical composition (classification)	0.004	0.060	0.020	0.753	**0.568**
Quality of alloy property prediction using cluster analysis (testing)	85.020	9.220	6.160	0.994	**0.989**
Quality of alloy property prediction using cluster analysis (validation)	107.570	10.370	6.190	0.993	**0.986**

result space ($R^2 = 0.986$). The results presented in Table 2 concern a two-stage classification procedure based on discretized variables of silumin's composition.

The application of the rough set theory in the analysis of data on property changes due to heat treatment of an experimental bronze alloy is another example of machine learning application [6]. The inference mechanism supporting the selection of appropriate heat treatment parameters and modifiers to obtain the desired bronze properties was based on the induction of rules and decision trees with algorithms using the rough set theory and also on the exploratory analysis of the research results, including statistical analysis and analysis of variance (ANOVA). The use of the tool in the form of rough sets solved the problem of the granularity of information, as well as the problem of the uncertainty and incompleteness of data under the conditions of discrete and point signals.

Machine learning tools were also found to be useful in the comparative analysis of material properties. Various machine learning tools were used to solve the problem of selecting material for the manufacture of a product with given mechanical properties. Classifier models allowed supporting technological decisions at the stage of designing foundry products, especially the types of Ductile Iron, and cast iron—ADI and nodular cast iron with carbides. The analysis was carried out in two ways. On the one hand, the classification problem was investigated based on raw property data, and on the other hand, an improvement introduced to this method by the use of a preliminary property-based cluster analysis, making the classification not only more coarse but also error-free, was studied. The solution obtained in this way combines the advantages of the DTs, ANN, SVM, and kNN.

These studies were expanded by research on the development of an exploratory analysis of microstructural data of compacted graphite iron (CGI) in order to obtain knowledge about the formation of ausferrite [6]. Rules have been developed to estimate the ausferrite content based on chemical composition. Models were developed using (1) statistical tools, regression—GLM, (2) regression boosted trees—BT, (3) random forest—RF, (4) artificial neural networks, (5) piecewise regression models—PR, (6) the CART algorithm, and (7) MARSplines. The procedure consisted of creating subsequent models on sequentially selected data sets. It can be compared to creating a random forest and the so-called "weak learners". Results obtained for individual models are shown in Table 3.

With the process of data filtering, the precision of models has changed. After removing the cases that were easy to learn (not containing ausferrite), the models

Table 3 The quality of prediction in iteratively reduced data sets

	MSE	MAE	RMSE	R	R^2
ANN(180)	22.9	2.9	4.8	0.99	0.99
ANN(72)	34.2	4.2	5.8	0.97	0.93
BT(180)	30.1	3.4	5.5	0.99	0.98
CART (pearlite submodel)	55.4	3.9	7.4	0.79	0.62
GLM(180)	48.7	3.3	7.0	0.99	0.97
MARS (microstructure based)	2.3	0.9	1.5	0.998	0.995
PR(180)	40.0	3.2	6.3	0.99	0.98
PR(72)	51.8	4.6	7.2	0.94	0.89
RF(180)	130.8	7.2	11.4	0.97	0.94

worsened the fit factors, but at the same time the accuracy in predicting actual contents improved (Table 3). In the boosted trees example, the change due to the reduction of the data set can be clearly seen. On the full set, the algorithm determined over 800 trees, yielding an error of 30, while on the reduced set, about 320 trees were enough, but the MSE error almost doubled. Multiple studies of the significance of the influence of individual factors in various regression models allowed us to select the most important variables influencing the content of ausferrite in various areas of parameter variability [7]. The research is an example of the use of several machine learning algorithms in the acquisition of knowledge about the CGI microstructure. The use of multiple algorithms, as well as the analysis using two scenarios, i.e., both classification based on raw data and previously done cluster analysis, allowed for a comparison of the results of these procedures. The results are summarized in Tables 4 and 5, wherefrom it clearly follows that earlier cluster analysis improves the quality of classification. Moreover, the study has shown that in situations where precision is more important than interpretability, artificial neural networks and the support vector machine method give much better results than decision trees.

The paper also presents an example of the use of experimental source data to build models forecasting the volumetric composition of individual components of the microstructure (ferrite, pearlite, carbides, martensite, ausferrite, and austenite) of

Table 4 Results of the use of several machine learning algorithms in microstructure classification

Methods	ANN	kNN	CART	CHAID	SVM
Fitness: χ^2	17.8	22.3	3.7	0.1	9.2
Fitness: G^2	43.8	32.5	17.0	2.1	26.8
Incompatibility (%)	13.8	48.0	50.5	73.7	42.0
Training (%)	86.2	–	31.6	46.4	90.9
Testing (%)	63.3	–	–	–	58.0
Validating (%)	60.0	41.2	52.1	52.1	82.9

Table 5 Results of the use of several machine learning algorithms in microstructure classification after clustering

Clusters	CART	SVM	ANN
Fitness: χ^2	3.1	0.21	0.12
Fitness: G^2	42.9	4.2	6.12
Incompatibility (%)	9.8	1.38	2.07
Training (%)	97.9	98.7	97.9
Testing (%)	–	99.9	99.9
Validating (%)	97.6	99.02	99.9

compacted graphite iron (CGI). The data set used to train the models was collected as a result of observation and measurements of the content of individual components of the CGI microstructure in relation to the content of individual alloy additives (molybdenum, nickel, and copper) introduced in different proportions for different thickness casting walls [7]. The hybrid neural-fuzzy algorithm was used to build the predictive model—ANFIS (Adaptive Neuro-Fuzzy Inference System) [8]. The research allowed to develop a fuzzy inference engine (FIS) containing 108 fuzzy rules of the SUGENO type in which the premises refer to the developed fuzzy sets, and the conclusions are functional dependencies between the adopted variables. Figure 2 shows the FIS model parameters (Inputs/Outputs) determined in the learning process using the ANFIS algorithm.

input	inputmf	rules
Cu, %		108 rules were generated in the model. The general form of the rule is presented in formula:
Ni, %		$$IF\left(x_1 \text{ is } A_1^i\right) \text{ and... and } (x_n \text{ is } A_n^i)$$ $$THEN\,(y=b^i)$$
Mo, %		The exemple of the rule:
thickness, mm		*IF(Cu is S) and (Ni is S) and (Mo is M) and (thickness is 3)* *THEN ausferrite = 822.8 Cu + 7.893 Ni - 79.92 Mo + 76.91 thickness -90.33*
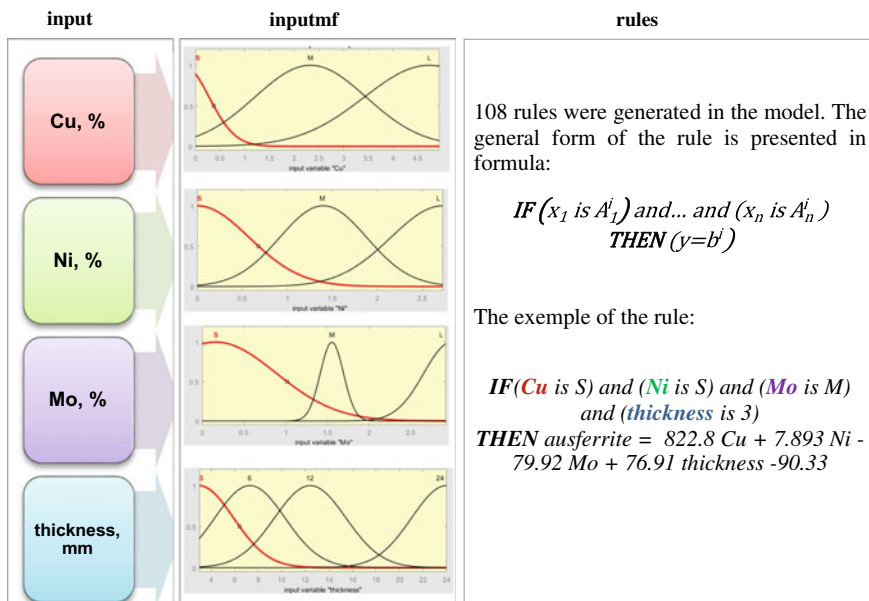
Fig. 2 The FIS model parameters (Inputs/Outputs) determined in the learning process using the ANFIS algorithm

The developed system allowed (with a training error of 5% and a testing error of about 9%) its users to predict the content of a selected component in the microstructure of compacted graphite iron. The developed model can be used in the selection of the chemical composition of compacted graphite iron (CGI) and, consequently liquidate the expenditure or reduce the cost on experimental studies aimed at demonstrating the possibility of producing cast iron with an ausferritic matrix microstructure.

The conducted research showed a very high potential of the adaptive neural-fuzzy inference system (ANFIS). Based on measurement data, it enables optimization (tuning) of fuzzy model parameters. The fuzzy models enable the generalization of the knowledge contained in the noisy measurement data and presenting it in a form that is understandable for a human.

5   Processes

Process models being a result of machine learning algorithms have a working pattern similar to the approach related to the study of material properties. In each case, the task of the models is, on the one hand, to enable forecasting (for a specific decision parameter) and, on the other hand, to assess how individual process input parameters change the output value (sensitivity analysis).

The use of decision trees was supported by research into the bolt production process within the production chain. Based on process data, it was possible to develop, for example, CART and CHAID decision tree models [6]. The analysis of the process data allowed for the assessment of the importance (the strength of influence) of the variables on the bolt quality class. The achievement was the generation of rules allowing the detection of possible production defects. The use of decision trees enabled interpretation of the model (as a whole or in fragments—individual rules) by a human. This is a very important aspect of data mining, and unfortunately, it cannot be done using other techniques. The transparency of models obtained with the help of decision tree induction is, however, often at the cost of reduced precision or overfitting of the model (taking into account insignificant deviations). However, studies have shown that this tool can be used in process control as a decision support with an accuracy of up to 94%.

The decision tree induction method was also used in the research on the rolling process of dual-phase steel strips in order to optimize it [9]. A solution is presented that allows studying the influence of individual process parameters (22 independent variables in the model, including charge temperature, inter-operation times for six passes, roll rotational speed, heat transfer coefficient, etc.) on the output parameters (rolling temperature in individual passes and grain size). The results of the statistical and exploratory analysis were compared with the results of the sensitivity analysis [10]. It has been shown that the application of the proposed techniques allows achieving the same results using only 1% of the data needed to perform the

sensitivity analysis by conventional computational techniques. This solution enables quick analysis, even in real time.

Another example of data-based modeling is molding sand moisture testing [11]. As part of this research, a method for improving the quality of the molding sand preparation process based on moisture prediction was developed. Exploratory research enabled discovering the relationship between the parameters responsible for the properties of molding sand. A two-step approach to prediction fitting (using both tree induction and cluster analysis) was developed and demonstrated to be effective in assessing the moisture content in molding sand. In this way, knowledge in the rule-based form was discovered, necessary in building knowledge bases for decision support systems in the diagnosis of molding sand bonding.

In the research on the analysis of the durability of forging tools, fuzzy logic and artificial neural networks were used to develop systems predicting the wear of forging tools for the selected die forging process [12–16]. Currently, the costs of forging tools constitute a significant share of the total costs of forging production, all research aimed at increasing the durability of forging tools, and reducing their share in the production costs is important. The developed system makes it possible to calculate the wear of the tool for the given values of its operating parameters and estimates the intensity of the occurrence [%] of typical mechanisms of tool destruction (i.e., thermo-mechanical fatigue, mechanical wear, abrasive wear, and plastic deformation). As source data, the results of many years of experimental research and numerical modeling were used, which concerned the impact of selected inputs factors of the forging process on the tool wear characteristics. These studies included, among others, the following issues: observation of the forging process performance, macroscopic analysis combined with scanning of the tool working surfaces, microhardness measurements, microroughness analysis, and numerical modeling of the forging process. The explanatory (inputs) variables were the number of forgings, pressures, temperature on selected tool surfaces, friction path, and type of protective layer applied on the tool. The explained (output) variables included the wear of the tool (geometric loss of material [mm]) and the percentage contribution of the main destructive mechanisms. In this study, many formal methods were considered. As a starting point, the results of many years of industrial research collected in the developed database were used. Considering the fact that the results of material tests and computer simulations are most often incomplete and may be burdened with errors (measurement and simulation), the analysis was carried out mainly with the use of formal methods, which in their nature allow modeling based on uncertain and incomplete data. The research was carried out using fuzzy logic [12], the ANFIS neuro-fuzzy algorithm [13], and the artificial neural networks [14–16]. The scheme of developed research for the system predicting the wear of forging tools is shown in Fig. 3.

The best fit and the smallest error were obtained when using artificial neural networks [14–16]. In the developed system works nine neural networks, five of them determine the value of the geometric loss of the material for tools working with different protective layers (Fig. 4a–e), and four networks define the intensity of the occurrence of destructive mechanisms (Fig. 5a–d).
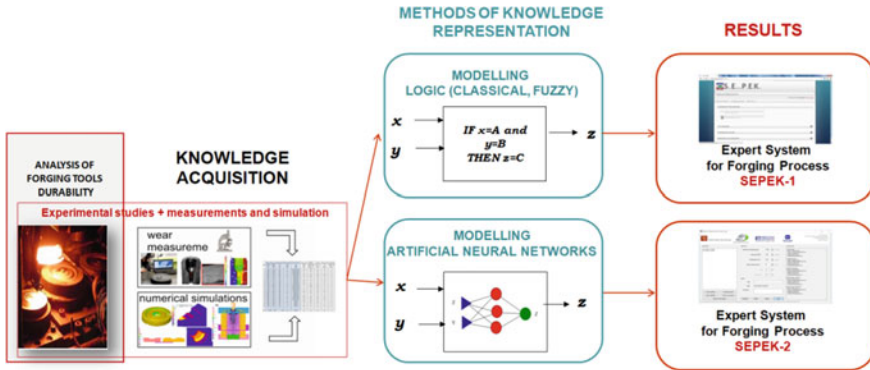
**Fig. 3** Scheme of studies developed to design a system that predicts the wear and tear of forging tools

The system developed with the use of artificial neural networks enables the analysis and prediction of the durability of forging dies with the identification of critical areas and mechanisms of destruction.

A system of this type is not only the source of valuable scientific knowledge but also, above all, a practical tool for process engineers and technologists working in the forging industry.

6 Formalization of Knowledge: Development of Knowledge Bases and Semantic Integration

In process modeling, material databases are often used. To make decisions and to predict the future in production processes, knowledge consisting not only of data but also, above, all of the technological skills is used. The works described earlier showed that knowledge about processes can also be acquired automatically with the use of machine learning. In order to fully use the potential of the applied methods of artificial intelligence, the acquired knowledge in the form of rules, models, or patterns should be saved, preserved, or codified. To make the preserved knowledge additionally understandable for humans and processable for machines, it is necessary to describe it properly.

Part of the authors' research work deals with the methods of knowledge formalization. The process of knowledge discovery and inference using machine learning models and the semantic techniques (ontology) to formalize knowledge and codify it in semantic knowledge bases was presented. The use of ontology development techniques, i.e., semantic modeling, can be presented on the example of ontology for the description of multi-scale models of thermomechanical alloy processing [17]. The semantic model, or ontology, enables the validation of multi-scale models by
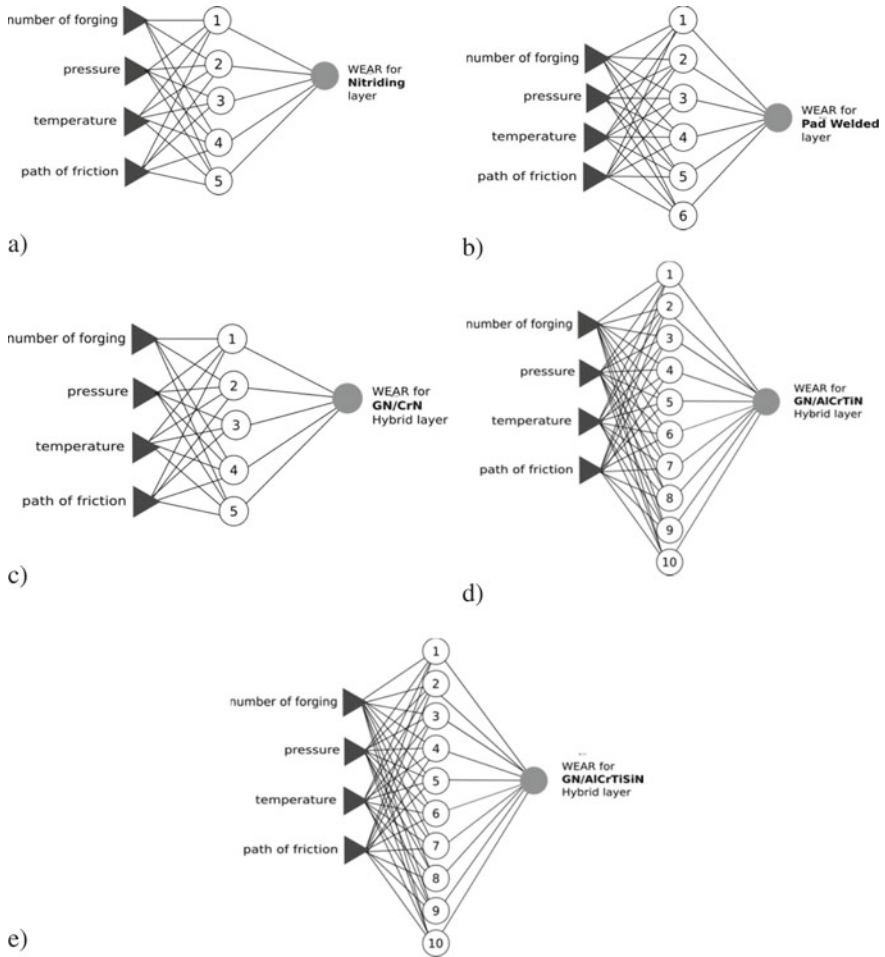
Fig. 4 A simplified list of artificial neural networks operating in the system—ANNs for loss of material (wear) for tools working with protective layers: **a** nitrided, **b** padded, **c** hybrid Cr/CrN, **d** hybrid Cr/AlCrTiN, and **e** hybrid Cr/AlCrTiSiN

examining the completeness of the input variables. A description of this type is also a tool for unambiguous communication between engineers and programmers.

A system was developed that enables automatic, ontology-based cataloging and substantive search of text documents from the repository of documents related to metal processing technology. This solution enables the creation of a semantic knowledge base in the field of metallurgy and metal processing with the use of acquisition and codification techniques of reusable computer artifacts generated from documents. The methodology includes a TF-iDF algorithm to determine the discriminant power of words, hidden LSI semantic indexing to group documents, creating groups of synonyms, and integration with ontology by assigning classes to thematic groups.
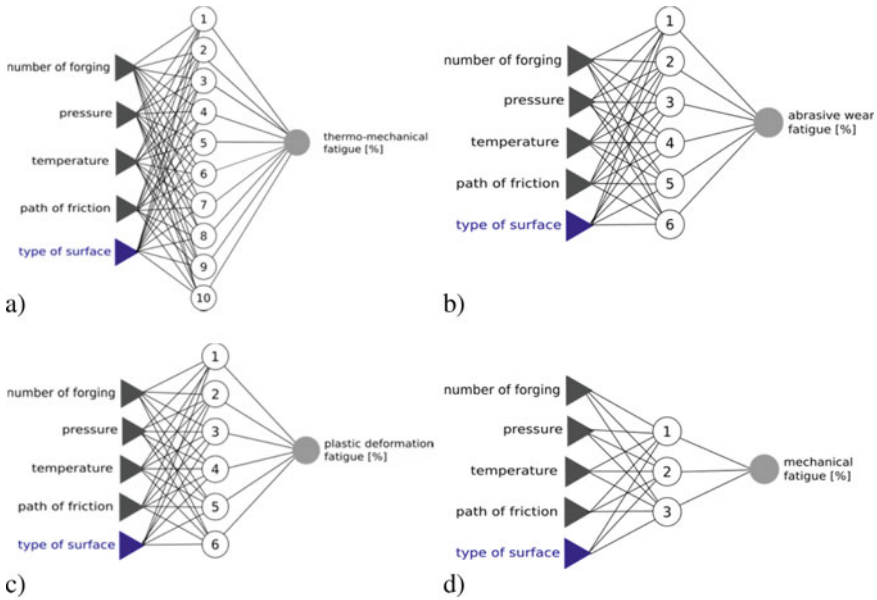
Fig. 5 A simplified list of artificial neural networks operating in the system—ANNs for destructive mechanisms: **a** thermo-mechanical fatigue, **b** abrasive wear, **c** plastic deformation, and **d** mechanical fatigue

The development of the system forced the process of knowledge formalization, which is an indispensable element in the creation of decision support systems. The solution consisted of the adaptation of classifiers based on the rough set theory and analysis of associations in the text mining of documents concerning the technology of production and processing of metals [18].

7 Conclusions

The developed methodology provides a universal tool for knowledge acquisition, offering, on the one hand, the precision of neural networks or support vector machines, and, on the other hand, the intuitiveness and transparency of decision trees, combined with the possibility of in-depth analysis of dependencies. The knowledge bases created in this way, formalized using the latest methods of knowledge representation, meet the needs of Industry 4.0 users and participants to the greatest possible extent. It has also been shown that they can be used in the production of various types of materials with very specific characteristics. The conducted research has demonstrated that it is possible to use the same methods of artificial intelligence (machine learning) to model various processes in the area of metal processing. Thanks to the conducted research, it was possible to propose a set of analytical tools that allow for

solving and modeling various, often complex, problems of metal processing, while acquiring knowledge about them in a form that is both human-understandable and machine-processable, which is a necessary condition to support the idea of Industry 4.0.

References

1. Hey, T., Tansley, S., Tolle, K. (2009). *The fourth paradigm: Data-intensive scientific discovery*. Microsoft Research, ISBN: 978-0-9825442-0-4.
2. Mueller, T., Kusne, A. G., & Ramprasad, R. (2016). Machine learning in materials science: Recent progress and emerging applications. *Reviews in Computational Chemistry, 29,* 186–273.
3. Bartók, A. P., Poelking, C., Bernstein, N., Kermode, J. R., Csányi, G., & Ceriotti, M. (2017). Machine learning unifies the modeling of materials and molecules. *Science Advances, 3*(12).
4. Butler, K. T., Davies, D. W., Cartwright, H., Isayev, O., & Walsh, A. (2018). Machine learning for molecular and materials science. *Nature, 559*(7715), 547–555.
5. Regulski, K., Wilk-Kołodziejczyk, D., Kluska-Nawarecka, S., Szymczak, T., Gumienny, G., & Jaskowiec, K. (2019). Multistage discretization and clustering in multivariable classification of the impact of alloying elements on properties of hypoeutectic silumin. *Archives of Civil and Mechanical Engineering, 19*(1), 114–126.
6. Regulski, K. (2020). Data mining and machine learning in aspects of acquiring knowledge about the production and processing of metals for the needs of Industry 4.0. *Hutnik 2020*(4). https://doi.org/10.15199/24.2020.4.3.
7. Mrzygłód, B., Gumienny, G., Wilk-Kołodziejczyk, D., et al. (2019). Application of selected artificial intelligence methods in a system predicting the microstructure of compacted graphite iron. *Journal of Materiels Engineering and Performance, 28,* 3894–3904. https://doi.org/10.1007/s11665-019-03932-4.
8. Jang, J.-S.R. (1993). ANFIS: adaptive-network-based fuzzy inference system. *IEEE Transactions on Systems Man and Cybernetics, 23*(3p.), 665–685. https://doi.org/10.1109/21.256541.
9. Regulski, K., Wilk-Kołodziejczyk, D., Szymczak, T., Gumienny, G., Gietka, T., Pirowski, Z., et al. (2019). Data mining methods for prediction of multi-component Al-Si alloy properties based on cooling curves. *Journal of Materials Engineering and Performance (JMEP), 28,* 7431–7444. https://doi.org/10.1007/s11665-019-04442-z.
10. Szeliga, D., Kusiak, J., & Rauch, Ł. (2012) Sensitivity analysis as support for design of hot rolling technology of dual phase steel strips. In: J. Kusiak, J. Majta, & D. Szeliga (Eds.), *Metal Forming 2012: Proceedings of the 14th International Conference on Metal Forming* (pp. 1275–1278). Weinheim: Wiley-VCH Verlag GmbH & Co. KGaA (Steel Research International).
11. Rauch, Ł., Kusiak, J., & Regulski, K. (2020). Artificial Intelligence in steel industry—From casting to final product. In: *The Metal Forming Conference MEFORM* (pp. 11–14). ISBN 978-3-86012-632-5.
12. Gronostajski, Z., Hawryluk, M., & Kaszuba, M., et al. (2016). The expert system supporting the assessment of the durability of forging tools. *The International Journal of Advanced Manufacturing Technology, 82*, 1973–1991. https://doi.org/10.1007/s00170-015-7522-3.
13. Hawryluk, M., Mrzygłód, B. (2016). Application of adaptive Neuro-Fuzzy Inference System (ANFIS) to predict the wear of forging tools. In: *Metal 2016: 25 International Conference on Metallurgy and Materials* (S. 90), May 2016, Brno, Czech Republic, Eu: list of abstracts. Ostrava: TANGER Ltd., cop. 2016. ISBN: 978-80-87294-66-6.

14. Mrzygłód, B., Hawryluk, M., Gronostajski, Z., Opaliński, A., Kaszuba, M., Polak, S., et al. (2018). Durability analysis of forging tools after different variants of surface treatment using a decision-support system based on artificial neural networks. *Archives of Civil And Mechanical Engineering, 18*(4), 1079–1091. https://doi.org/10.1016/j.acme.2018.02.010.

15. Hawryluk, M., & Mrzyglod, B. (2018). A system of analysis and prediction of the loss of forging tool material applying artificial neural networks. *Journal of Mining and Metallurgy, Section B: Metallurgy, 54*(3), 323–337. https://doi.org/10.2298/JMMB180417023H.

16. Mrzygłód, B., Hawryluk, M., Janik, M., et al. (2020). Sensitivity analysis of the artificial neural networks in a system for durability prediction of forging tools to forgings made of C45 steel. *International Journal of Advanced Manufacturing Technology, 109,* 1385–1395. https://doi.org/10.1007/s00170-020-05641-y.

17. Macioł, P., & Regulski, K. (2016). Development of semantic description for multiscale models of thermo-mechanical treatment of metal alloys. *The Journal of The Minerals JOM, 68,* 2082–2088.

18. Regulski, K. (2017). Formalization of technological knowledge in the field of metallurgy using document classification tools supported with semantic techniques. *Archives of Metallurgy and Materials, 62*(2), 715–720.

Accurate, Real-Time Replication of Governing Equations of Physical Systems with Transpose CNNs — for Industry 4.0 and Digital Twins

Hritik Narayan and Arya K. Bhattacharya

Abstract The Universal Approximation Theorem provides the theoretical basis for perceptron-like architectures to represent the functionality of complicated mathematical functions to any desired degree of accuracy. Among the most complex of such functions are the governing equations of physical processes like the Navier-Stokes and Maxwell's equations. Accurate representation of complex physical phenomena through numerical simulation of such governing equations is a challenge, and it is a concomitant challenge for Artificial Neural Networks (ANNs) to learn the functionality of these equations from data generated from such simulations. There is an intense practical value of such analysis—most sub-processes in the industry are describable by approximate versions of these equations, their solution in real time will enable significantly more informative and precise monitoring, control and prognostics of running processes. Arrays of sensors installed at the physical boundaries of process domains can provide the crucial inputs to ANN-like architectures that can transform these isolated values into detailed field conditions. Provided these ANN-like mechanisms can exhibit the following properties—that they respond in process real time, their solutions are nearly as accurate as of the offline numerical simulations of the governing equations from which they learn the functional relationships, and importantly, they can map a few score inputs into around two orders of magnitude more number of outputs. The development of such architectures will open entirely new vistas of application of ANNs to modern industry. Here, we present convolutional-NN-like architectures based primarily on transpose convolutions and other design features that satisfy all the three crucial properties. These are demonstrated on two different application domains of the reduced Navier–Stokes equations, containing high nonlinearities and abrupt discontinuities including shocks.

Keywords Convolutional Neural Network · Neural networks · Transpose convolutions · Real-time process monitoring · Process simulations · Differential equations governing physical processes

H. Narayan · A. K. Bhattacharya (✉)
Ecole Centrale School of Engineering, Mahindra University, Hyderabad 500043, India

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

1   Introduction

Governing equations of complex physical processes are most often expressed as sets of coupled Partial Differential Equations in a few defining variables—as, for example, the Navier–Stokes equations of Fluid Mechanics [25], Maxwell's equations of Electromagnetics [18], constitutive equations of Solid Mechanics, etc. The complexity of these equations inhibits closed-form solutions even for the simplest geometries, and they have to be solved numerically, typically on a grid in the field that conforms to spatial boundaries. At these boundaries, the known conditions, e.g. values of some of the variables, are provided from which the field solution at grid points is extracted through simulation [3, 8] using appropriate numerical schemes. These simulations are invariably time-consuming, ruling out accurate solutions in anything close to real time.

According to the Universal Approximation Theorem, proven independently by Cybenko [7] and Hornick et al. [14], any type of Artificial Neural Network with at least one hidden layer and nonlinear activation functions is able to replicate any arbitrarily complicated mathematical function to a desired degree of accuracy. The hidden implication is that sufficient data extracted from the mathematical function are available for the Artificial Neural Network (ANN) to learn this functionality.

Combining the above aspects, if accurate numerical simulations can be attained for arbitrarily complex physical phenomena described by corresponding governing equations, then in principle the resulting simulation data can be used to train variants of ANNs to represent the functionalities encapsulated in these equations. Needless to say, both these steps exemplify significant open technological challenges. One may also question the utilitarian value of the second step.

For many application domains and scenarios, the accurate solution of complex physical problems in real time can lead to quantum improvements at technocommercial levels. For example, in Industry 4.0 which represents the complete manifestation of IoT (Internet of Things) in process and manufacturing industry [17, 24], every individual subprocess in the production chain needs to be monitored and controlled for highest performance. Considering any such individual sub-process, the values of relevant variables defining the process can be known at its physical boundaries from sensors, but the values at the field interior remain unknown in real time. If these could be extracted, monitoring and control would be significantly improved, the immediate upstream and downstream sub-processes in the chain would be more accurately defined, and productive efficiency of the entire chain would be enhanced. Numerical solution of the governing equations of the physics of the sub-process is the most relevant mechanism for extracting the variable values at the field interior, but as mentioned above, they are unattainable in process real times.

This brings us to the utilitarian value of the mentioned second step. Simulations carried out on the governing equations representing the concerned physical processes to relevant degrees of accuracy, can be used to train ANN variants, which may then, in production phases, take values from sensors to generate detailed conditions in the field interior. This can be achieved in practice provided the ANN variants being

designed, trained, and satisfy three crucial conditions. These are: (a) any forward pass through the ANN should be executed in process real times, ideally a few milliseconds, (b) the accuracy levels should be very high with representative metrics like say Mean Absolute Percentage Error (MAPE) of the order of 2% or less, and (c) the number of inputs will be (approximately) the number of boundary grid points in the original simulation while the number of outputs shall be the number of field grid points, and the latter will invariably be at least *one order of magnitude higher* in number.

Condition (c) is the biggest challenge as most (nearly all) ANN variants in diverse domains of application are designed with the number of outputs less than or equal to the number of inputs. In most cases, outputs are significantly less than the number of inputs. The combination of conditions (b) and (c) makes this a profound challenge.

We have used Convolutional Neural Network (CNN) architectures based on Transpose Convolutions and other design patterns to upscale from a small number of inputs to large outputs, larger by two orders of magnitude. This is done by maintaining error levels (MAPE) at around 2% and less, demonstrated on two complex, highly nonlinear problems with strong field discontinuities from the domain of fluid mechanics. Having in mind requirement (a), one of our design priorities has been to keep the overall number of parameters small. An aspect common to nearly all CNN architectures—one or more fully connected layer(s) preceding the output layer— create an explosion in the number of parameters but yet are an essential element needed for accuracy attainment. Our architectures completely eliminate this fully connected layer without compromising on accuracy.

The technical contributions of this work include:

- Synthesizing Transpose Convolutional architectures with specific design features that can learn the functionality of systems with outputs two orders of magnitude more than the number of inputs.
- Architectures that avoid any fully connected prefinal layer and, hence, require about an order of magnitude less number of parameters as compared with conventional ANNs or CNNs, to reproduce comparable levels of accuracy.
- Combining the above, completing the pipeline for exploiting advances under Industry 4.0 to realize instantaneous and accurate monitoring, prognostics, control and optimization of industrial processes and Digital Twins.

The rest of the paper is structured as follows. Section 2 discusses some contemporary-related developments. Section 3 presents two application domains that demonstrate both the relevance and conceptual validity of these architectures. Section 4 describes the architectural principles and their proof of concept on the selected application domains. Finally, the conclusions are presented.

2   Related Work and Significance in Industry

In this section, we present a short overview of current related developments in the area of replicating or augmenting numerical solutions of governing equations of physical processes with Machine Learning-based techniques. This is followed by a brief explanation of Transpose Convolutions and fractional strides and then a discussion of some crucial nuances for creating real-time mappings from process sensors to field interior solutions in industrial scenarios.

2.1   Related Developments

There has been a significant amount of work right from the early days of Machine Learning to use different types of ANNs to simulate the equations governing different physical processes; Adie [1] and Brunton et al. [6] provide fairly comprehensive reviews of applications of Machine Learning techniques in different areas of computational science and engineering.

A set of active applications of ANNs is in the area of turbulence closure in Computational Fluid Dynamics. The complete Navier–Stokes' equations (DNS) do not need turbulence closure but are extremely resource- and time-intensive to solve. The next one and two levels of approximations—namely, Large Eddy Simulation (LES) and Reynolds Averaged Navier–Stokes' (RANS)—split the velocity components into a mean and a fluctuating (turbulent) part implying that the number of parameters increase by three, which need to be solved through "closure" models that add to the original set of equations. ANNs have been used in different ways to provide these closures [2, 11]. Sinai et al. [23] learn these models as ANNs trained on accurate DNS solutions. Duraisamy et al. [12] provide a comprehensive review of the state of the art.

Kim et al. [15] have used Generative Models trained on accurate but small sets of CFD simulation data to synthesize plausible and divergence-free 2-D and 3-D velocities. Xie et al. [29] developed a Generative Adversarial Network to learn spatial and temporal dynamics of turbulent flows, from limited data that, interestingly, are taken from only one-time step.

There are a set of techniques that apply the Koopman Operator [16, 19] in different ways for the resolution of unsteady flow characteristics. Morton et al. [20] have applied ANN models based on the Koopman operator to learn the forced and unforced dynamics of airflow over a cylinder directly from CFD data, and then used these learning-based approaches for controller design to manipulate the flow characteristics aft of the cylinder.

Brunton et al. [5] followed a novel approach to "discover" governing equations of physical systems from data, by considering equations to be composed of differential terms taken from a pool that may be extracted using Machine Learning techniques while maintaining an optimal level of complexity.
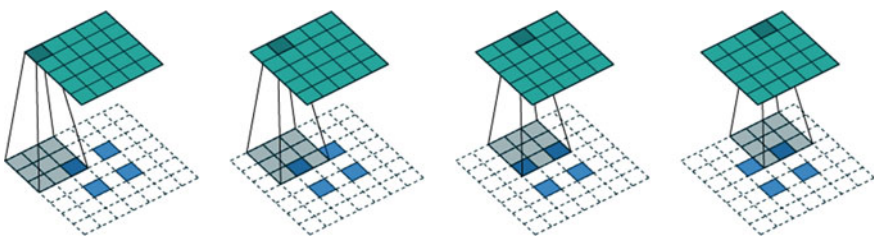
2.2   Transpose Convolutions

In a normal CNN, the data flow between any two "activation maps" are in the direction of input (of the network) to output, usually accompanied by a shrinkage of the area of map. Hypothetically, if data flow would be in the opposite direction, then this would be called a "deconvolution" or more correctly "transpose convolution". This hypothetical concept just helps in understanding that transpose convolutions enable expansion of the map area.

Figure 1 shows what can be called a transpose of a convolution, which may be viewed as a standard convolution with a filter of size 3 (both dimensions equal) moving with a stride of 2 over the green-colored upper image segment of size 5, resulting in a map of size 2 (blue-colored lower image segment) by using formula [21]

$$\text{new\_size} = \frac{\text{old\_size} + \text{pad\_1} + \text{pad\_2} - \text{filter\_size}}{\text{stride}} + 1. \tag{1}$$

Alternately, this can be viewed as a transpose of the above standard convolution, when mapping in the upward direction from blue to green segments when the two zero paddings are of size 2, stride is 1, and importantly, the size (of blue segment) has been inflated from 2 to 3 by adding one layer of zero internal paddings. Insertion of these numbers in (1) gives new_size as 5, which is the size of resultant map.

The filter's stride of one under conditions of one internal zero padding implies that two strides of filter are needed to move from an actual nontrivial pixel to the next. Hence, this can be actually viewed as half-a-stride, and, thus, the term "fractionally strided convolutions". One may note that this is reciprocal of the length of stride in the forward convolution. This work uses transpose convolutions and experimented with internal paddings with fractionally strided filter movements on certain layers.

Fig. 1 Illustrating the concepts of transpose convolutions and fractional strides This figure reproduced from Dimoulin and Visin [9] (in public domain)

2.3 Real-Time Mapping from Sensors to Field Interior Values

As stated earlier, one of the objectives of this work is to demonstrate the working principle for a pipeline mechanism that picks up variable values from sensors located at the physical boundaries of a process, uses them as inputs to an ANN-like architecture, and computes the variable values at field grid points as outputs—all within real-time cycles.

However, the sample data generated from simulations of the governing equations of process physics will have the boundary values at the boundary grid points (of the simulation), and it is not possible that the sensor locations in the physical system will be exactly colocated with these boundary grid points. Hence, an intermediate step is needed between the generation of simulated samples and their usage as training data for the ANN, where the variable values from the boundary grid points are interpolated onto the precise locations of the sensors, and these interpolated sensor-located variable values are the ones that will serve as the inputs for training of the ANN.

It follows from the above that the ANN is designed to receive the sensor values as inputs and generate the solution at field interior grid points as outputs. The number of sensors will be significantly less than the number of boundary grid points, which serves to amplify the challenge of factor (c) discussed in Sect. 1.

Figure 2 illustrates the type of scenarios we are trying to model. It is a view of a symmetric half of a wide section of the primary cooling mold of the continuous casting process of steel manufacturing [26, 27]. Here, molten steel is passed through a cooling mold with heat being extracted continuously through the copper-plated mold boundaries using a fine network of pipes with flowing water, which are embedded in the copper plate. Liquid steel (yellow zone) solidifies as it approaches the copper plates, and the layer touching (nearly), the plates is solid but soft (red). The governing equations in the mold interior represent a coupling of processes from fluid mechanics, solidification, and heat transfer; their simulations attainable only offline and quite complex [28, 30]. However, there is an array of thermocouples (temperature sensors) embedded in the copper plates, which provide continuous readings of temperatures (not visible in Fig. 2). The real-time temperature variations manifest all internal process variations within the mold.

Offline numerical simulations of the internal process considering a spectrum of conditions will yield data that can be used to train ANN-like architectures. The simulations can be performed on grids that extend from the mold boundaries (copper plates) to the field interior. Temperatures are one of the parameters of simulation. Temperatures at boundary grid points can be interpolated onto the sensor locations, and the ANN training will take these sensor-located temperatures as input and one or more parameter (like temperatures, velocities, pressures) values at the field grid points as outputs. Upon training, the ANN can be put in production mode, when the scanned values from the sensors can be mapped almost instantaneously into computed values throughout the field interior—creating a breakthrough in monitoring, control and optimization of operations.

Fig. 2 Conditions inside and at boundaries of the continuous casting process of steel manufacturing, figure reproduced from Thomas [26], with permission. More details of the continuous casting process may be found at https://en.wikipedia.org/wiki/Continuous_casting
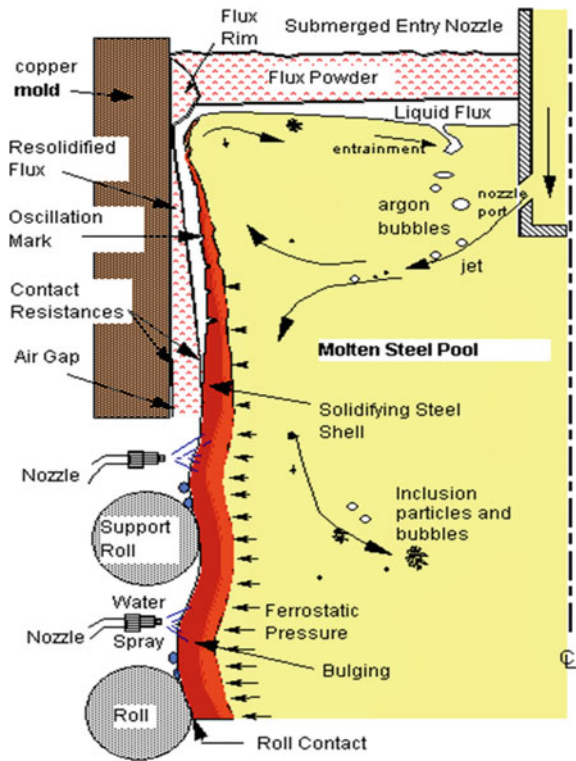
The same philosophy discussed above on the specific example of a continuous casting plant for steel *can be applied in multiple scenarios in diverse industries and processes.*

3 Application Domains for Demonstration of Concepts

Two application domains are considered where data generated from numerical simulations of the relevant governing equations of the underlying physical processes are used for training our CNN architectures, as discussed in prior sections. Both are from the field of fluid mechanics, representing highly nonlinear phenomena and are selected to possess-specific characteristics that together enable demonstration of all the special features of CNN representation that are the distinctive aspects of this development.

3.1 Compressible Potential Flow Over a Flat Plate at Incidence

Starting with the Navier–Stokes' equations and successively neglecting viscosity and field vorticity, one may arrive at the full potential equation [4, 13], which is written as

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \rho \nabla \phi = 0, \tag{2}$$

where ρ and ϕ represent density and velocity potential, respectively. The steady form of (2) can be represented as a Poisson equation

$$\nabla^2 \phi = -\frac{\nabla \rho \cdot \nabla \phi}{\rho} \tag{3}$$

with density expressed as

$$\rho = \left[1 + \frac{\gamma - 1}{2} M_\infty^2 \{ 1 - \nabla \phi \cdot \nabla \phi \} \right]^{\frac{1}{\gamma - 1}} \tag{4}$$

subscript ∞ refers to conditions at freestream, M denotes Mach number, γ the ratio of specific heats. The velocity $v = \nabla \phi$, and density ρ, have been normalized with respect to their freestream values V_∞ and ρ_∞.

Equations (3) and (4) represent a coupled nonlinear system. They are converted to their different forms and solved for flow over a flat plate at various angles of incidence α and freestream Mach M_∞ using the grid scheme that is illustrated in Fig. 3. The center of the grid is the flat plate shown in red with exaggerated thickness; actual geometrical thickness is effectively zero. Around the plate is a rectangular grid with x-axis-parallel j-lines and y-axis-parallel i-lines; i and j are the index numbers of the lines and (i, j) denote the indices of intersection points in the field at which the potential ϕ is computed in the simulations. There are a total of 3498 such points in the field.

It can be seen that the plate extends from $i = 17$ to 37 and is located at $j = 29$ and 30; the latter two lines are merged and in principle correspond to plate upper and lower surfaces. There are a total of 42 points on the surfaces where the boundary conditions are applied for making the numerical simulations and later, as we shall see, serve as the inputs to the CNN that is trained from the data generated by these simulations. Note that the CNN computes the field values, totaling 3498, as its outputs.

A series of numerical simulations are performed to generate a set of data samples for training of the CNN. A traverse is made over six values of M_∞ from 0.1 to 0.6, at intervals of 0.1 (Mach 1 has a ballpark value of 330 m/s). At each value of Mach, the freestream direction, i.e. the value of α, is swept over the range $-10°$ to $+10°$
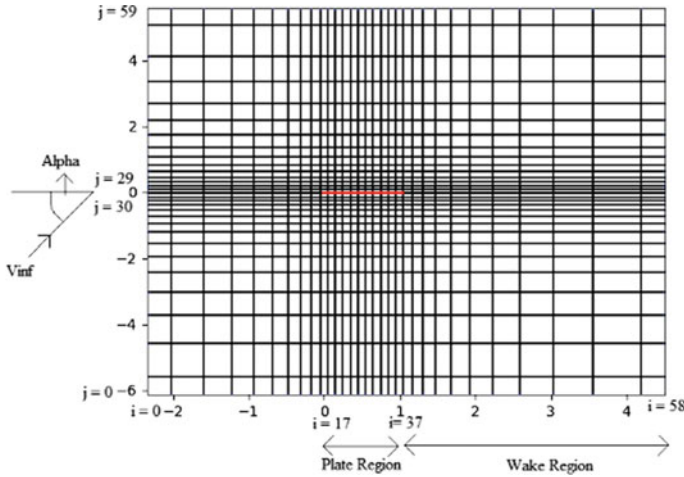
Fig. 3 Rectangular grid for simulation of incident flow over a flat plate. Alpha (α) and Vinf (V_∞) represent the angle of incidence and velocity of the freestream flow relative to the flat plate (shown in red). The orthogonal grid lines represent the mesh for numerical simulation

at intervals of $0.1°$, giving a total of 201 values. Thus, a total number of generated samples at all speeds are 1206, which will serve as the training and validation data for the CNN.

This application domain represents complex, nonlinear flow patterns where complexity increases sharply with M_∞ and $|\alpha|$. We have stopped just short of combinations that generate supersonic pockets and shock formations as the isentropic assumption of the full potential equation is not strictly valid under these conditions, though they can still be modeled in approximation using various numerical schemes [4, 13]. The objective of this application is to demonstrate one of the core capabilities of our CNN architectures, namely, computing outputs that are two orders of magnitude larger than inputs, with high accuracy and in real time.

3.2 Strong Shocks and Axial Drag Resolution over ONERA-M6 Wing

We selected this case because it comprises some very specific features quite unlike that of any other training and validation sample data used in other machine learning applications, and very relevant to the CNN architectures and concomitant functionalities we have in mind. This relates to an aircraft wing designed specifically to serve as a benchmark for developments on numerical simulations of the Navier–Stokes' equations, with challenging features like a double-shock on the upper surface of the same section and consequent boundary layer separations manifested in sharp changes in the skin-friction drag. This wing was tested in Schmitt and Charpin [22].
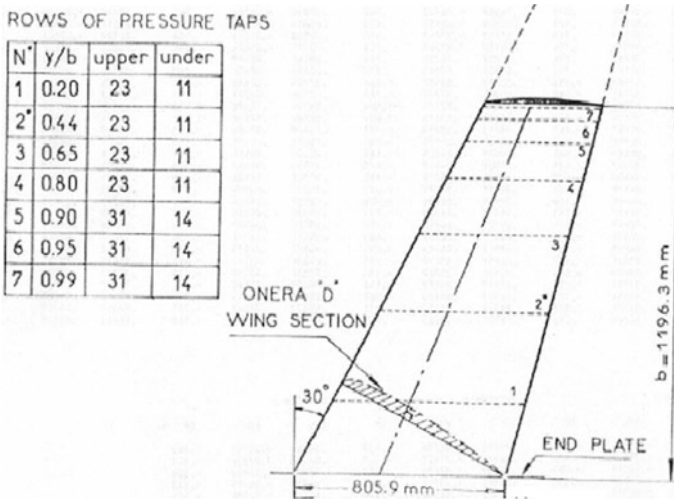
Fig. 4 Layout and instrumentation of the ONERA M6 wing, from fig. B1-1 of Schmitt and Charpin [22], in public domain

The major aspects of the design are shown in Fig. 4; we will try to explain some aspects of the core aerodynamics.

Pressure sensors are placed at seven wing sections located (*y/b* also termed as η) as shown in the table in Fig. 4, where *b* denotes the wing span and *y* extends from '0' at wing root to 1 at tip. There are 34 sensors at each of the first four sections, and 45 at the last three, distributed between upper and lower surfaces as shown in the table. At a flow case of $M_\infty = 0.84$ and $\alpha = 3.06°$, strong double shocks (severe discontinuities in process variables) are formed at most of the sections, resulting in boundary layer separations and steep variations in skin friction drag. These may be seen in Diskin et al. [10] and also in https://turbmodels.larc.nasa.gov/onerawingnumerics_val_sa.html, where the black symbols denote sensor readings of pressure coefficients C_p) and the pink lines denote results of CFD simulations. The simulations are from a specific class of turbulence models on the full Navier–Stokes' equations (see https://cfl3d.larc.nasa.gov/); the mathematical relations between skin friction coefficient $C_{f,x}$ and the pressure coefficient C_p are extremely complex and encapsulate a spectrum of physical phenomena.

We have created training and test samples taking the sensor measurements obtained from https://www.grc.nasa.gov/WWW/wind/valid/m6wing/m6wing.html as inputs and the CFL3D simulations, obtained from here[1], as outputs. At each section, there are 33 (sensor value) inputs and 368 outputs. We have taken two sets of outputs at each section, for C_p and for $C_{f,x}$, and created two separate sample data sets for two different CNNs. Importantly, we treat each section itself as a training sample, consider the section at $\eta = 0.65$ as a test sample, and remove the sample

[1] https://turbmodels.larc.nasa.gov/Onerawingnumerics_val/SA/CFL3D_OM6_A3p06_CPCF.dat.

at $\eta = 0.99$ because the flow features at the tip are very different from the rest. That leaves us with only five samples of training data which is grossly insufficient. We partially augment this by creating three new interpolated sections at $\eta = 0.32$, 0.55, and 0.72, linearly from neighboring sections. Using these eight samples, we investigate whether it is possible to create CNNs with 33 inputs and 368 outputs each, which can capture the functionality embedded in highly complex, nonlinear, discontinuous relationships with high accuracy.

There is yet another dimension to this application domain. This is an exact representation of the practical utility challenge (discussed in Sect. 2.3), namely, to pick up data from a handful of sensors capturing a specific process parameter, and see whether this can be used in a specifically architected CNN to reproduce a range of different parameters at points distributed over the field interior that are at least an order (of magnitude) more in number, accurately and in real time. The parameters are related through highly complex governing equations which are simulated to generate limited training data for the CNN.

4   Principles of CNN Architectures for Relevant Application Domains

One may recall the three properties desired of CNNs that will satisfy crucial performance requirements in the considered technical domains—that they shall map from process inputs to outputs one or two orders of magnitude more in number, shall exhibit high accuracy, and shall execute well within real times of the process (and, hence, use a minimum number of parameters). Here, we highlight the principles followed in designing CNN architectures that satisfy these requirements and remain valid across application domains.

4.1   Architectural Principles

For the class of technical problems that we aim to attend, invariably output sizes are about the same or more than input sizes in all considered dimensions (axes). Accordingly, we use transpose convolutions with external paddings of zeros and fairly large filter sizes to gradually, layer by layer, build up the size. Furthermore, the shape of the output map is likely to be quite different from the shape of the input map—manifested in significant variance between the aspect ratios of the two. Again, we use varying external zero paddings and filter sizes along the different axes to gradually, across layers, transform the input shape to that of the output. On occasions, at some of the layers especially those close to the output, we retain the shape and size across two or more activation maps with the objective of building up the total number of parameters for attaining desired accuracy levels.

An important consideration in our design has been the *complete elimination of fully connected layers*, especially those located immediately preceding the output, as these tend to very significantly increase the total number of parameters. However, they also enhance the accuracy of solution as these fully connected layers ensure that every pixel in the first input layer influences every single node in the output layer. One may recall that in typical CNN architectures, *a pixel has a cone of influence* gradually expanding across layers, the rate of expansion is proportional to filter sizes, but that does not guarantee that each input pixel influences every single output node. This is taken care of by the fully connected last layer. To eliminate fully connected layers while ensuring accuracy, we have used deliberately large filter sizes that enable faster expansion of the zone of influence of every input node to extend over the totality of nodes across the output map.

It is pertinent to note here that we converged at these above architectural design principles through a series of experiments performed on alternative architectures. Almost all these experiments were performed on the problem of flow over a flat plate, and we briefly highlight some of these.

The simpler, linear version of the flat plate problem is obtained by setting density to a constant (i.e. incompressible flows), thus automatically removing Eq. (4) and converting (3) to the Laplace equation

$$\nabla^2 \phi = 0 \tag{5}$$

was first solved numerically to generate data of exactly the same structure described in Sect. 3.1. This was used to train a conventional feedforward neural network architecture shown in Fig. 5. This work is reported in Sahil and Bhattacharya [21]. The input and output layers have 42 and 3498 nodes, respectively. There are two hidden
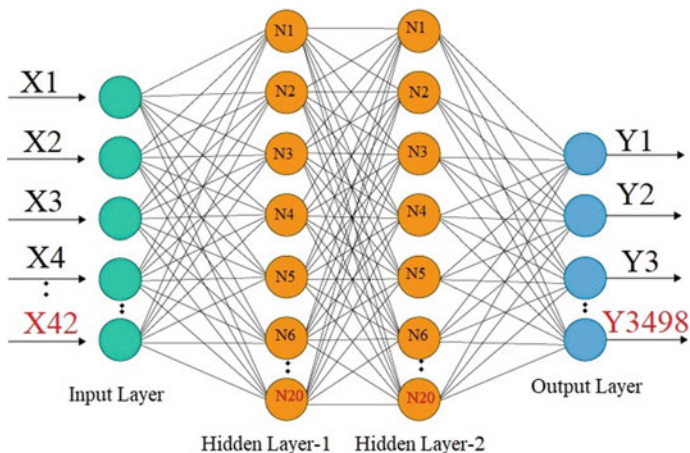
Fig. 5 Schematic of conventional ANN structure. Note that a number of outputs are two orders more than the number of inputs

layers each with 20 nodes, and all hidden and output layer nodes use the tanh(.) activation function. The number of parameters in this architecture is about 76,000. The obtained Mean Absolute Percentage Error (MAPE) was 1.2%.

Next, the full nonlinear potential equation was simulated to generate data of exactly similar structure. This was used to train the identical ANN architecture shown in Fig. 5. We could obtain a MAPE of 2.2%. The increase in error manifests the greater complexity of the underlying equation, especially the transition from modeling a linear to a nonlinear function.

The next set of investigations were performed on one-dimensional CNN architectures, where the 42 inputs, 3498 outputs, intermediate layer activation maps, filters and paddings were all in a single dimension. These were again on the linearized version of the full potential equation, used as a baseline for testing alternative designs. The best performing architecture is shown in Fig. 6, which used two intermediate zero paddings and fractional strides of 1/3, and importantly, a fully connected semi-final layer, which increased the number of free parameters to nearly 10 million. This of course was a violation of our design goals, but the purpose at this experimental stage was to explore whether CNNs can at all meet our aims. We obtained a MAPE of 2.3%, which was worse than our fully connected architecture. Removal of the pre-final fully connected layer significantly reduced the number of parameters but worsened convergence and error levels. At this point, we stopped our experimentation with 1-D CNNs and moved into 2-D architectures.

Within 2-D designs, we investigated various alternative architectural patterns including fractional strides, max-pooling and sequences of convolutional layers,
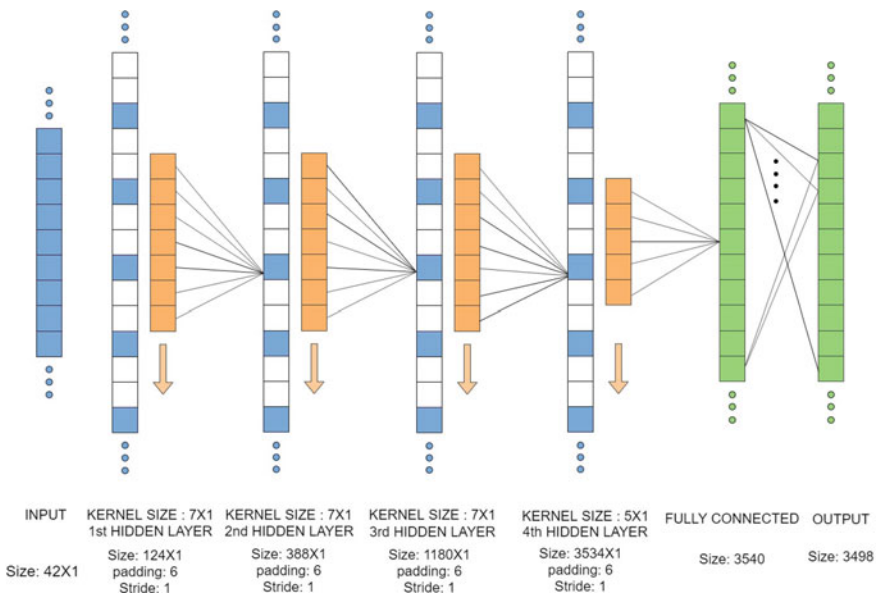

Fig. 6 One-dimension CNN architecture with a pre-final fully connected layer

but ultimately these led to sub-optimal performance compared with that obtained following the principles described earlier in this section. The successful design principles are reflected in Tables 1 and 2, which provide the architectural details of the two CNNs, where the first attends to the case of potential flow over a flat plate, and the second to skin friction drag and other variables for transonic flow over the ONERA M6 wing.

Here, we briefly describe the structure of Tables 1 and 2 that define the architectures of the CNNs representing the flat plate and the M6-wing. The rows represent the layers in sequence, the first being the input layer. In the first column, TCn denotes transpose convolution layer number n. The second column provides the two dimensions of the input. The next column provides the external zero padding applied on the two axes, separated by a semicolon, and the size of padding on the two sides of each axis. The filter dimensions are shown in the next column. The last column provides

Table 1 CNN architecture for full potential flow over flat plate

Layer	Input	External padding	Filter size	Output
TC1	**(2, 21)**	(4,0), (4,0)	(5,1)	(6,21)
TC2	(6, 21)	(9, 0), (9, 0)	(10, 1)	(15, 21)
TC3	(15, 21)	(14, 7), (14, 7)	(15, 8)	(29, 28)
TC4	(29, 28)	(15, 15), (14, 14)	(30, 30)	(29, 28)
TC5	(29, 28)	(15, 15), (14, 14)	(30, 30)	(29, 28)
TC6	(29, 28)	(15, 15), (15, 15)	(16, 16)	(44, 43)
TC7	(44, 43)	(21, 21), (22, 22)	(44, 44)	(44, 43)
TC8	(44, 43)	(16, 16), (16, 16)	(17, 17)	(60, 59)
TC9	(60, 59)	(29, 29), (30, 29)	(60, 59)	(60, 59)
TC10	(60, 59)	(29, 29), (30, 29)	(60, 59)	(60, 59)
TC11	(60, 59)	(29, 29), (30, 29)	(60, 59)	**(60, 59)**

Table 2 CNN architecture for transonic flow over Onera M6 wing

Layer	Input	External padding	Filter size	Output
TC1	**(3, 11)**	(0, 9), (0, 9)	(1, 10)	(3, 20)
TC2	(3, 20)	(0, 19), (0, 19)	(1, 20)	(3, 39)
TC3	(3, 39)	(0, 39), (0, 39)	(1, 40)	(3, 78)
TC4	(3, 78)	(1, 49), (1, 49)	(2, 50)	(4, 127)
TC5	(4, 127)	(2, 63), (1, 63)	(4, 127)	(4, 127)
TC6	(4, 127)	(0, 59, (0, 59)	(1, 60)	(4, 186)
C1	(4, 186)	–	(3, 3)	(2, 184)
TC7	(2, 184)	(1, 91), (0, 92)	(2, 184)	(2, 184)
TC8	(2, 184)	(1, 91), (0, 92)	(2, 184)	(2, 184)
TC9	(2, 184)	(1, 91), (0, 92)	(2, 184)	**(2, 184)**

output dimensions, which again appear as the input dimensions in the next layer. Note that the CNN in Table 1 maintains one channel throughout the network, whereas the CNN in Table 2 uses two channels that are combined in the final layer. The relation between the output and input along each axial direction is given by Eq. (1), which also incorporates padding, filter size and stride. We have consistently used a stride of 1, avoiding fractional strides. Related information on transposed convolutions and fractional strides may be found in Dimoulin and Visin [9].

4.2 Presentation of Results

The CNN represented in Table 1 is used to predict test cases for specific pairs of M_∞ and α for the case of potential flow over a flat plate. This has a total of only 15,047 parameters and the accuracy attained after 15,000 epochs is a MAPE (Mean Absolute Percentage Error) value of 2.4% on the validation cases.

This CNN uses Leaky-ReLU ($\alpha = 0.05$) as activation at all layers except the last layer where sigmoid is used. Adam optimizer with *amsgrad* is used to train the algorithm, with $lr = 0.005$ and $\beta_1 = 0.9$, $\beta_2 = 0.999$. The training algorithm cycles between mini-batch sizes of 20 and 100 for every 500 epochs. No dropout techniques are employed.

Results are presented for a case of $M_\infty = 0.6$ and $\alpha = 9°$ in Fig. 7. This is a strongly nonlinear, near-supersonic flow case. A few grid points next to the leading edge upper surface have local Mach number at just about 1, i.e. the supersonic threshold. The numerical simulation technique cannot prima facie handle supersonic pockets; however, at this flow condition, the decomposed x- and y-components of velocity individually remain subsonic. Figures 7a and b present the x-component of velocity from the original simulation and its replication by the CNN, respectively, in the form of contour plots. Figures 7c and d do likewise for the y-component of velocity. While the CNN representation is not perfect, it has captured all major characteristics of the field velocity components especially those close to the plate boundaries. Small perturbations are seen in regions in the field interior.

Figures 8a and b show the resolution of flow parameters over the ONERA M6 wing by the CNN shown in Table 2, with only 6667 parameters. The CNN learns the relationship between 33 values of pressure coefficients at sensor locations distributed over eight sections on the wing surface (33 sensors at each section) and 368 values of pressure coefficients and skin friction drag coefficients each distributed on the surface at corresponding sections obtained from simulations performed on the program CFL3D (see Diskin et al. [10]).

At the considered flow case of $M_\infty = 0.84$ and $\alpha = 3.06°$, large supersonic pockets are formed on the wing upper surface and field, which terminate in strong dual shocks. These shocks trigger boundary layer separation with very sharp dips in skin friction drag coefficient. These phenomena involve strong nonlinear couplings between multiple flow features and resolving them through numerical simulations is a challenge. Figures 8a and b compare CNN outputs with CFL3D simulations
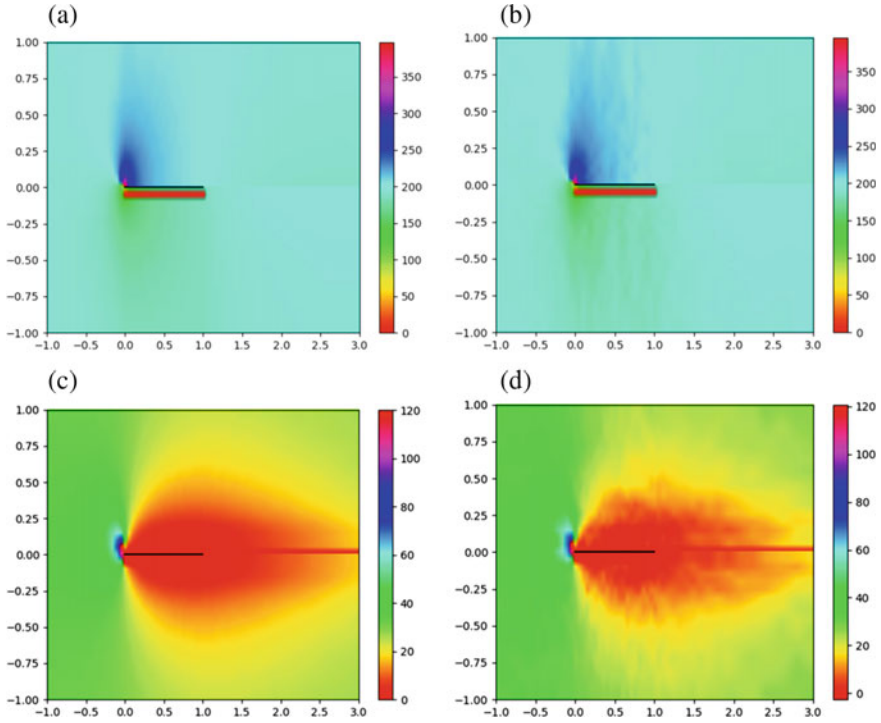
Fig. 7 **a** v_x, x-component of velocity, simulated using our CFD simulation, for the case $M_\infty = 0.6$ (i.e. V_∞ approximately 198 m/s) and $\alpha = 9°$. The x- and y-axes represent distances normalized against the length of flat plate, with origin at its leading edge. Color contours represent velocities in meters/sec, according to the vertical color bar at right. **b** v_x, x-component of velocity, predicted using our CNN. **c** v_y, y-component of velocity, simulated using our CFD simulation. **d** v_y, y-component of velocity, predicted using our CNN. **a–d** Results for full potential flow over a flat plate at incidence

for pressure—and skin friction drag coefficients, respectively, at a test section η = 0.65, that is not used for training. Figure 8a also shows the sensor values at discrete locations. Superficially, it may appear that Fig. 8a interpolates pressures from sensor inputs to finely distributed points on the surface. *This is not the case.* CNN outputs (as functions of inputs) are learnt from CFL3D simulations, which solve from fundamental governing equations and are unaware of sensor readings and use them purely for validation. The CNN learns the functional relationship between sensor readings of pressures only and the simulated values of pressures (Fig. 8a) and skin friction (Fig. 8b).

One may appreciate that the functional relationship between pressures from two different sources is easier to learn than that between pressures and skin friction drag, which involve multiple cause-and-effect-of-variables-through-equations relationships. While Fig. 8a shows fairly accurate representation, Fig. 8b reveals that the sharp dips at the two boundary layer separation points—representing extreme nonlinear phenomena—are not captured very precisely. It is pertinent to recall here

Fig. 8 a C_p predicted versus CFL3D simulations and sensor measurements. **b** $C_{f,x}$ predicted vs CFL3D simulations. **a–b** Results at section $\eta = 0.65$ on ONERA M6 wing
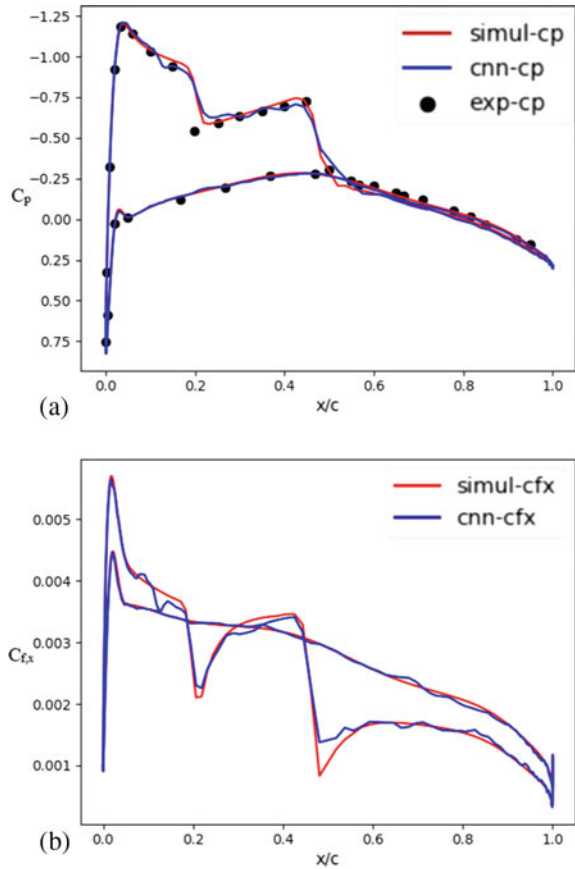
that *we use only eight training data samples for 6667 parameters*—and issues of under fitting (bias) are expected to arise. That is another facet of the challenge for this case. The small wiggles seen on the CNN solutions are possibly a consequence of this.

Apart from architecture and accuracy, the third crucial aspect is computation time for a single forward pass from input to output. On an implementation using *tensorflow*-1.15.0 and *keras*-2.3.1, on an NVIDIA DGX-1 system utilizing a single *Tesla* V100 GPU, a single sample run of the FP CNN takes 1 ms, while a single sample run of the M6 CNN takes 3 ms. These are well within process real-time cycles in industrial production or associated computations like in Digital Twins.

5 Conclusions

This work embodies innovations in two different domains, Artificial Intelligence specifically Neural Network Architectures, and Industry 4.0 and Digital Twins, developments in the first facilitating the second.

Related to the architecture and design of Convolutional Neural Networks, it has achieved an output-to-input-size ratio of about two orders of magnitude, which is a significant advance in this technical domain. This is obtained using transpose convolutions coupled with certain specific principles of design.

Importantly, the above is attained while completely eliminating a fully connected pre-final layer, which drags down computational performance by blowing up the number of parameters, thus, reducing computing times by about an order of magnitude with minimal compromise on accuracy. This by itself may be considered a breakthrough in this field.

The above developments make this an enabler of a very desirable feature in Industry 4.0, including Digital Twins, namely, informed and precise real-time monitoring, prognostics and control of complex systems and processes. Inputs of specific parameters acquired from sensors placed at the physical boundaries of such processes can be used to obtain a comprehensive picture of the instantaneous state of the total process.

References

1. Adie, J. (2018). Deep learning for computational science and engineering. Retrieved September 30, 2020, from http://on-demand.gputechconf.com/gtc/2018/presentation/S8242-Yang-Juntao-paper.pdf.
2. Barone, M. F., et al. (2017). Machine learning models of errors in large eddy simulation predictions of surface pressure fluctuations. In *AIAA 2017-3979, in 47th AIAA Fluid Dynamics Conference.*
3. Bartoli, A. L. D., Andreis, G. S., & Pereira, F. N. (2015). *Modeling and simulation of reactive flows.* Elsevier. ISBN: 978-0-12-802974-9.
4. Bhattacharya, A. K., & Arora, N. L. (1994, February). A hybrid integral equation—Finite volume scheme for transonic potential flow over complex configurations. *Aeronautical Journal*, 34–48.
5. Brunton, S. L., Proctor, J. L., & Kutz, J. N. (2016). Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences, 113*(15), 3932–3937. https://doi.org/10.1073/pnas.1517384113.
6. Brunton, S. L., Noack, B., & Koumoutsakos, P. (2020). Machine learning for fluid mechanics. *Annual Review of Fluid Mechanics, 52,* 477–508. https://doi.org/10.1146/annurev-fluid-010719-060214.
7. Cybenko, G. (1989). Approximations by superposition of a Sigmoidal function. *Mathematics of Control, Signals, and Systems, 2,* 303–314.

8. Dede, E. M., Lee, J., & Nomura, T. (2014). *Multiphysics simulation.* London: Springer. https://doi.org/10.1007/978-1-4471-5640-6.

9. Dimoulin, V., & Visin, F. (2018, January). A guide to convolution arithmetic for deep learning. arXiv:1603.07285v2, https://arxiv.org/abs/1603.07285, Retrieved September 30, 2020.

10. Diskin, B., et al. (2018). Grid convergence for three dimensional benchmark turbulent flows. In AIAA Paper 2018-1102, 2018 AIAA Aerospace Sciences Meeting.

11. Duraisamy, K., Zhang, Z. J., & Singh, A. P. (2015). New approaches in turbulence and transition modeling using data-driven techniques. In *AIAA 2015-1284, 53rd AIAA Aerospace Sciences Meeting.*

12. Duraisamy, K, Iaccarino, G., & Xiao, H. (2019, January). Turbulence modelling in the age of data. *Annual Review of Fluid Mechanics*, *51*, 357–377. https://doi.org/10.1146/annurev-fluid-010518-040547.

13. Holst, T. L. (1995, July). Numerical solution of the full potential equation using a chimera grid approach. In NASA-TM-110360.

14. Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks, 2,* 359–366.

15. Kim, B., et al. (2019). Deep fluids: A generative network for parameterized fluid simulations. In P. Alliez, & F. Pellacini (Eds.), *EUROGRAHICS.* Wiley.

16. Koopman, B. O. (1931). Hamiltonian systems and transformation in Hilbert space. *Proceedings of the National Academy of Sciences, 17*(5), 315–318.

17. Lade, P., & Srinivasan, S. (2017, May–June) Manufacturing analytics and industrial internet of things. *IEEE Intelligent Systems*, 74–79.

18. Maxwell, J. C. (1865). A dynamical theory of the electromagnetic field. *Philosophical Transactions of the Royal Society, 165,* 459–512.

19. Mezic, I. (2013). Analysis of fluid flows via spectral properties of the Koopman operator. *Annual Review of Fluid Mechanics, 45,* 357–378. https://doi.org/10.1146/annurev-fluid-011212-140652.

20. Morton, J., et al. (2018). Deep dynamical modeling and control of unsteady fluid flows. In *32nd Conference on Neural Information Processing Systems, NeurIPS.*

21. Sahil, K., & Bhattacharya, A. K. (2019). Accurate replication of simulations of governing equations of processes in Industry 4.0 environments with ANNs for enhanced monitoring and control. In *2019 IEEE Symposium Series on Computational Intelligence.* https://doi.org/10.1109/ssci44817.2019.9003058.

22. Schmitt, V., & Charpin, F. (1979, May). Pressure distributions on the ONERA-M6-Wing at transonic mach numbers. *Experimental Data Base for Computer Program Assessment.* Report of the Fluid Dynamics Panel Working Group 04, AGARD-AR-138.

23. Sinai, Y. B., et al. (2019, July). Learning data-driven discretizations for partial differential equations. *Proceedings of the National Academy of Sciences, 116*(31). https://doi.org/10.1073/pnas.1814058116.

24. Stankovic, J. A. (2014). Research directions for the Internet of Things. *IEEE Internet of Things Journal, 1*(1), 3–9.

25. Stokes, G. G. (1843). On some cases of fluid motion. *Transactions of the Cambridge Philosophical Society, 8,* 105–137.

26. Thomas, B. G. (2002). Modelling of the continuous casting of steel—Past, present and future. *J. Metallurgical and Materials Trans, 33B,* 795–812.

27. Thomas, B. G. (2005). Modeling of continuous casting defects related to mold fluid flow. In 3rd International Congress Science & Technology Steelmaking, Charlotte, NC, 9–12 May 2005 (pp. 847–861). Warrendale, PA: AIST.

29. Xie, Y., et al. (2018, August). tempoGAN: A temporally coherent, volumetric GAN for super resolution fluid flow. *ACM Transactions on Graphics, 37*(4). https://doi.org/10.1145/3197517.3201304.

28. Yang, H., Vanka, S. P., & Thomas, B. G. (2019). Mathematical modeling of multiphase flow in steel continuous casting. *ISIJ International, 59*(6), 956–972. https://doi.org/10.2355/isijinternational.ISIJINT-2018-743.

30. Zappula, M. L. S., et al. (2020, April). Multiphysics modelling of continuous casting of stainless steel. *Journal of Materials Processing Technology, 278.* https://doi.org/10.1016/j.jmatprotec.2019.116469.

Deep Learning in Vision-Based Automated Inspection: Current State and Future Prospects

R. Senthilnathan

Abstract Deep learning has influenced almost all major domains of science, technology and engineering fields. The deep learning revolution started with the groundbreaking accuracy obtained in a computer vision problem. Machine vision-based inspection has been one of the pioneering applications of computer vision for industrial applications. The adoption of deep learning for machine vision applications took some time, and the current adoption rate though is satisfactory, it is observed that there is still a long way to go. The contents of the chapter is intended for beginners and managers who are evaluating application of deep learning techniques for vision-based automated inspection. This chapter presents detailed insights of merits and limitations of deep learning techniques for automated inspection tasks, especially in comparison to non-deep learning route. The various aspects of commissioning such as important pitfalls to be cautious about before choosing deep learning, deep learning software, deep learning hardware, types of deep learning networks and their inferences and potential applications in various types of industries are also discussed.

Keywords Automated inspection · Deep learning · Computer vision · Machine vision · Vision hardware · Deep neural networks · Vision software

1 Introduction to Computer Vision and Machine Vision

Vision is one of the most important senses in biological beings such as mammals which have directed and accelerated the progress of evolution for millions of years. The amount of information perceived from human vision as a sense is so vast and complex that nature evolved a separate portion in the brain called the visual cortex. The vision system in humans is so complex that there is a decentralized strategy for processing namely at eye-level, optical nerves and the visual cortex. The process of mimicking biological vision with cameras and computers is the primary goal of

R. Senthilnathan (✉)
Department of Mechatronics Engineering, SRM Institute of Science and Technology, Kattankulathur, India
e-mail: senthilr4@srmist.edu.in

the field of computer vision. Applying computer vision to industries that perform some form of manufacturing operations is called machine vision. Though machine vision is being applied to known environments in contrast to computer vision, it has its challenges to deal with. Most of these challenges arise from the requirements of productivity and robustness for 24×7 performance. The applications of vision in industries may be classified broadly into the following four types, popularly abbreviated as GIGI:

- *Gauging*: Deals with geometric measurements, locations, the distance between two or more points, etc.
- *Identification*: Deals with attributing an object with a name based on appearance, reading 2-D data-codes, characteristics printed on parts, etc.
- *Guidance*: Deals with locating the position, orientation, get the information required for automata such as robots, etc.
- *Inspection*: Deals with finding flaws, absence of crucial components of a system, broken parts, and other irregularities, etc.

This chapter deals with the various aspects of using deep learning for machine vision applications. Special emphasis is on the automated inspection using images and videos containing information about the real world. The other three types of applications are touched upon at a shallow level for the sake of completeness.

1.1 Vision-Based Automated Inspection

Inspection is an important integral part of any manufacturing system. The inspection aims at rejecting nonconforming parts and assuring good quality parts. Traditional labour-intensive visual inspection has been successfully replaced by machine vision systems in industries since the last three decades. A machine vision system for inspection detects any of the features presented in Fig. 1.

2 Introduction to Deep Learning

Artificial intelligence and specifically deep learning have gained tremendous momentum in this decade and are getting more and more powerful. Deep learning is a branch in machine learning which has dominated this decade in terms of the new applications in various fields of science, engineering and technology which were traditionally considered difficult if not impossible. Deep learning uses neural networks with multiple hidden layers; hence the word deep. A simple neural network is illustrated in Fig. 2 with two hidden layers.

Beyond the word deep, the significant potential of deep learning lies in its ability to do feature representation learning which is a contrasting difference against machine
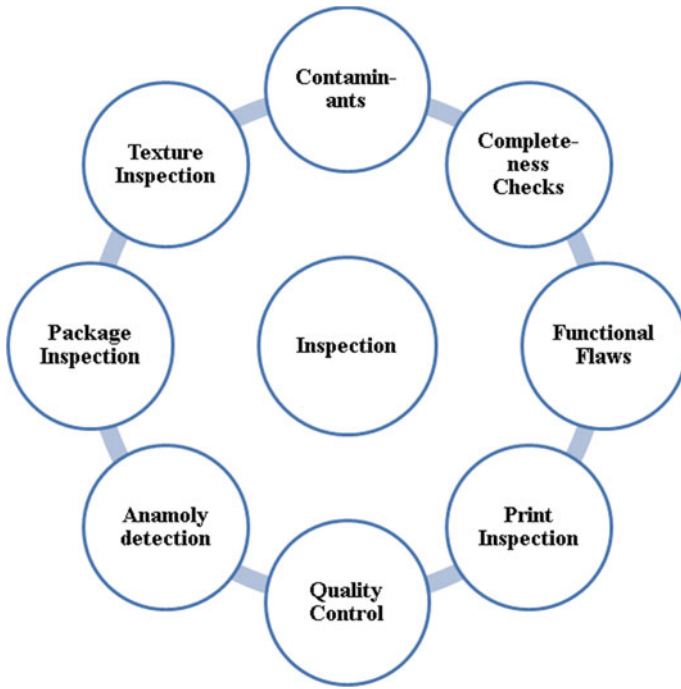
Fig. 1 Types of percepts in vision-based automated inspection

Fig. 2 Illustration of a simple two hidden layer network
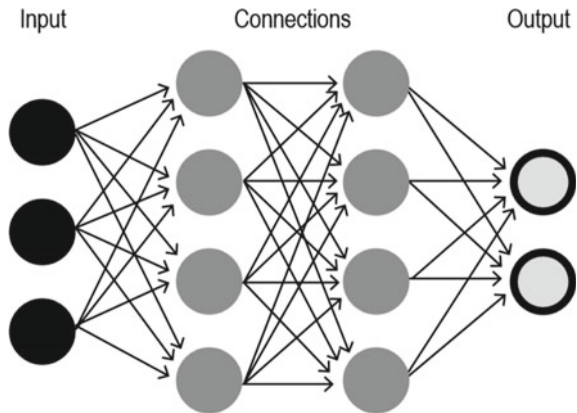
learning approaches which requires features to be extracted using tailor-made algo-rithms. Feature representation learning allows the deep learning networks to build the right set of features based on the available data and the task it is expected to perform. This one ability of deep learning is the primary reason for its exceptional accuracy in a variety of applications which were traditionally considered difficult if

not impossible. Features based machine learning approaches were not as accurate as deep learning since the features extracted by the engineers are limited to knowledge of the mathematics invented by humans. For example, there is no proper feature that can distinguish between a scratch (a defect) and an ascribed line that is intentionally made on the surface of an object. Some of the features which deep learning methods model during the learning process are still to be defined with existing mathematics. This unbounded ability gives the deep learning approach a huge edge in applications such as autonomous driving which is a completely unstructured problem that requires distinguishable representations of the various objects in the scene for creating a classification boundary.

Deep learning algorithms (in general all machine learning algorithms) may be classified into the following five types based on the learning process:

- *Supervised learning*: Maps input data with known labels annotated by humans.
- *Unsupervised learning*: Learns the inherent pattern in the unlabelled data.
- *Semi-supervised learning:* Uses partially labelled input data with the majority being unlabelled.
- *Reinforcement learning*: Learns how to maximize its goals based on the input data.
- *Transfer learning*: Reuses a model that was learnt from different sets of data for a all new dataset or fine-tuning of the existing.

In the spectrum of applications, the service applications were quick to adopt deep learning in fields such as driverless cars, surveillance, video analytics, web applications, etc. The industries were very sceptical about its functionality in the first half of the decade starting from 2012–13 during which most important milestones in vision-related tasks such as image classification were accomplished (*The Alexnet breakthrough* [1]). Industrial impressions about deep learning may be broadly categorized into any one of the following:

- Deep learning is an industry disruptor
- Deep learning is the next step in the technological evolutionary chain.

2.1 Traditional Vision Versus Deep Learning for Vision-Based Inspection

Conventional machine vision may be seen in two manifestations. One, completely rule-based methods and secondly manual features based machine learning involved methods. Rule-based methods do not employ any machine learning methods and use hard codes to extract features, analyse and make decisions based on them. The rules for classification are created by the human based on the multiple trials and tweaks of data and code respectively. Classical machine learning machine vision methods use methods such as support vector machines, classical shallow neural networks, etc. where the pattern is learned by an algorithm based on the features tailor-made by the

human. In either case, the features from the data (images and videos) are extracted by humans and further processed to make decisions. The rules for classification are created by the learning algorithm based on the input features extracted by the mathematics drafted by humans.

The table presents a rule-centric difference between the traditional vision- and deep learning-aided machine vision techniques (Table 1).

Deep learning has a fundamental difference in the way rules are constructed wherein the features required to construct rules and the rules themselves are generated by the network itself. This paradigm shift in the approach defines the advantages and limitations of deep learning.

From the perspective of inspection, the important points of comparison between deep learning with human vision and traditional machine vision may be observed in the Table 2.

The Fig. 3 captures the advantage of using deep learning over traditional machine vision in terms of the complexity of the applications and the amount of investment required [2].

Just like traditional machine vision, deep learning is not a technology that can solve 'anything anywhere'. Some of the key questions to be asked before deploying deep learning include the following.

Where to use it?

Deep learning is not a measuring tool. In industries, one of the most important types of application is metrology and gauging. Since deep learning networks learn the pattern from data, their performance is very good for semantic inferences rather than outright mechanical measurements such as form, geometrical primitives such as diameter, area, etc. Applications which use the appearance of the objects as the primary feature of interest for identification, inspection, etc. are the ones which might hugely benefit from deep learning way of solving.

Table 1 Human vision versus deep learning

The aspect of comparison where deep learning is an advantage	Human vision
Consistency	Not good. Mainly attributed to multiple inspectors across working shifts and fatigue of individual
Reliability	Not good. Scaling up the activity of inspection and the ability to reproduce good results in other lines is unreliable
Speed	Not good. Though subjective, deep learning can identify defects in milliseconds thus supporting high-speed inspection and large throughput use cases

Table 2 Traditional machine vision versus deep learning

The aspect of comparison where deep learning is an advantage	Traditional machine vision
Complexity of applications	Complex inspection and classification may be difficult due to the limitations of the existing mathematics to create boundaries of classification in hyper-space
Configurability	Quick prototyping and development are difficult due to the requirement of modifications of rules and other parameters of various algorithms which need to be tweaked manually
Invariances	Extremely prone to errors when variations in imaging conditions such as illumination, contrast, camera noise, transformations of objects of interest, background clutter, etc. occur. Lacks the generalization capability of deep learning
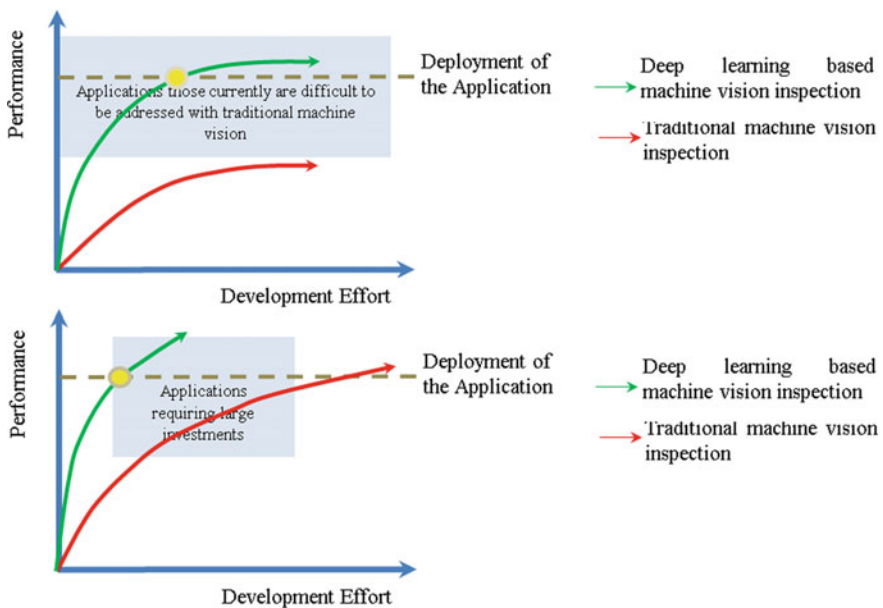
Fig. 3 Traditional Machine Vision Inspection versus deep learning-aided Inspection

What should be data for training deep learning networks?

Deep learning is a data-driven approach. Data is the primary requirement. Though there is a popular saying that 'more the data fed to deep learning, better is their performance', there is an underlying condition that needs to be satisfied with the

statement to be true. Data collected for training deep learning networks must be able to capture all real-world possibilities in terms of shapes, sizes, rigid-transformations, similarity transformations, background changes (*if any),* colour, relative movements, occlusions (*if applicable*), etc. The data collected for vision-based automated inspection systems must contain all the attributes of the scene in the right proportions as applicable for any given application. The word proportion is related to the probability of occurrence of the listed attributes in the real-world condition. Higher weight to a specific attribute may create undue bias during training. One of the effective ways to address this problem is to generate the data required for training from the actual process in industries rather than laboratory mock-ups. This ensures high fidelity of data in terms of how well it represents real-world information both in terms of quantity and quality. The other side of the data apart from the acquisition part is the annotation especially in a supervised learning framework, which is the majority of those used in machine vision systems. Eliminating any form of bias is one of the non-trivial processes involved in getting the data to a training-ready form.

How to train the system?

Data alone is not sufficient. Deep learning networks learn from data that are good representations of the real-world information subjected thoroughly to the training procedure adopted. The systematic approaches adopted in training are one of the primary reasons for the current state of the art performance of deep learning. Such procedures are ways of exploiting the full potential of deep learning philosophy. These include the strategy to create train-validation-test sets, hyperparameter tuning, early stopping, scheduling of hyperparameters such as learning rate during training, etc. In a vast majority of cases of deep learning networks used in applications such as classification and detection, the final few error percentage is reduced only due to the training strategy adopted as compared to the native mathematics constituting the network architecture.

How fast the deep learning algorithms run?

Deep learning networks used for machine vision is large. The very nature of deep learning networks, immaterial to the kind of data they are handling, is the large number of computations involved. This can be seen in multiple manifestations namely—number of layers of neurons, number of neurons, types of computations involved (e.g. simple linear combinations, 2-D convolutions, etc.), type of learning procedure (such as back-propagation), etc. Though the training time is not a real concern for industrial deployment, the time taken for generating an inference for any given data is extremely vital in influencing the productivity of the industry. During the inference, most deep learning networks only involve a forward-propagation which involves hundreds of thousands of matrix operations. This process demands a lot of computational resources. This requires fast and real-time computers with the ability to parallelize the processing. With the advent of modern GPUs, this is very much possible with many custom software and hardware products customized for accelerating deep learning inferences. Despite these advances, speeds greater than 500 parts-per-minute are very difficult to achieve with large deep learning networks used
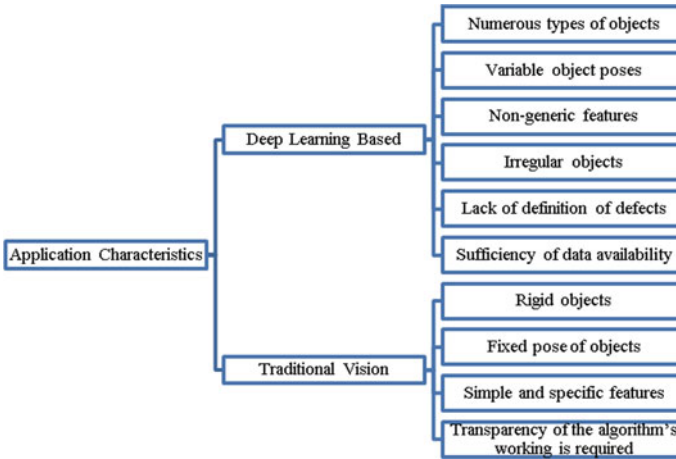
Fig. 4 Characteristics of applications

for detection. As a reference, such a milestone was achieved a decade ago in classical machine vision.

Can the industry trust the results from a deep learning algorithm?

Deep learning is received with scepticism. With a maturity level of almost more than five years of industry adoption, deep learning is still being viewed as a technology that may flaw. The main reason for this is the fact that there is no clear explanation for how the deep learning neural networks arrive at a certain inference. This is often viewed as a lack of transparency.

Given all the points of comparison, it must be understood with all the benefits of deep learning there are still many fundamental limitations. This also means that traditional machine vision is not obsolete. They still have potential applications where deep learning is either not viable due to lack of data generation for training and in principle; the application may not be solvable using data-driven approaches, e.g. geometric measurements. The Fig. 4 presents the various characteristics of applications that can employ deep learning and those which can be satisfactorily solved using the traditional machine vision route.

These application characteristics of traditional vision and deep learning may be applicable for a variety of applications in industries which are illustrated in Fig. 5 [3].

3   Deep Learning Methods in Machine Vision

One way of classification of deep learning networks is based on the nature of annotation concerning the data and inference generated by the network. In general, vision
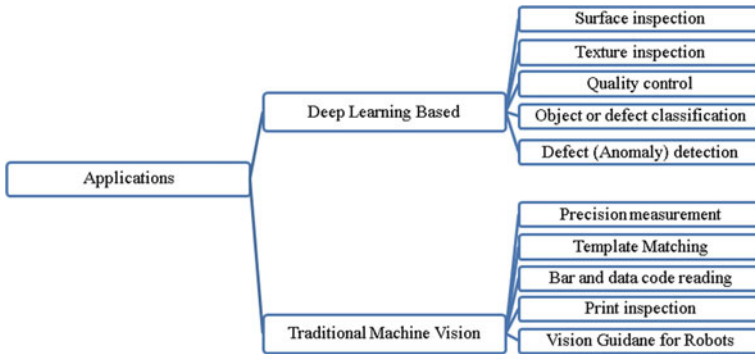
Fig. 5 Types of applications

tasks may involve the following problems while dealing with the various properties that need to be extracted about the objects in the image:

- *Image classification*: Associating an image to some object class without locating it.
- *Object Recognition*: Knowing what type of objects is available in the image.
- *Object Localization*: Location of the objects.
- *Object Detection*: Recognition + Localization.
- *Object/Image/Semantic Segmentation*: Associating every pixel to some object class.

Among the above mentioned tasks, the following are the various types of methods based on deep learning used in industrial applications:

Image Classification

Image classification is the simplest task among the types of method. In a supervised learning framework, the image classification problem contains an image with often one object labelled (generally images of the same object are saved in a single directory, wherein the directory name would aid in creating labels). This saves time compared to other methods where image-level intervention is required during the annotation process. In the case of using image classification for automated inspection, the sample images for good ones and the various types of defective parts' images are grouped in individual folders. Some of the popular deep learning networks for the image classification task include AlexNet [1], VGG [4], Googlenet [5] and ResNet [6].

Object Detection

Object detection localizes trained object classes inside an image and identifies them through a surrounding polygon (commonly a rectangular bounding box). Object detection is required in scenarios where multiple objects are available in the field of

view of the camera. Touching, overlapping and occluding parts must also be separated in multi-object inspection use cases. Object detection inference for surface inspection on metallic parts is a typical example under this category [7]. Various types of surface defects are the object classes that are trained with images and corresponding annotations to obtain such outcomes. The annotations are images and the corresponding coordinates of the polygon. It can be appreciated that the annotation process is extremely time-consuming as compared to image classification. Some of the popular deep learning networks for the image classification task include RCNN [8], Fast RCNN [9], Faster RCNN [10], Mask RCNN [11], Single Shot Detector [12] and YOLO [13] series of networks.

Semantic Segmentation

While object detection locates defects in an inspection process within a polygon, with semantic image segmentation, defect classes can be localized with pixel-level accuracy. Semantic image segmentation allocates a label for each pixel, with background and regions which are not of interest sharing a class label. The labelling process is way more time-consuming compared to object detection. Such networks are best labelled with special software tools available which can bring down the labelling time significantly lower. Some of the popular deep neural networks for semantic segmentation task includes U-Net [14], Feature Pyramid Network (FPN) [15] and DeepLab family of networks [16].

Apart from the three types, sometimes a fourth type of method which may use deep learning networks is called anomaly detection. Anomaly detection may be realized as image classification, segmentation or detection problem wherein multiple good products and very few bad parts are available for training. Such a scenario is generally the norm in automated surface inspection wherein detection and segmentation of defects is a requirement. The ability to solve some applications with fewer data is a special attribute of anomaly detection. Anomaly detection sometimes may be carried out without labels required if the images are of high quality in the context of capturing the differences between the defect regions and good regions significantly. In such cases, only a few images are required sometimes as low as 20 images of good parts. This makes the training fast, and hence enabling quick prototyping of the application code.

3.1 Challenges in Adopting Deep Learning for Machine Vision

Some of the primary challenges in adopting deep learning for machine vision are listed below.

Proprietary Issues: Unlike service applications such as the driverless car community where the massive datasets of annotated images and videos are made available open for access by the technical community, industrial application sample has to be

Fig. 6 Deep learning enablers
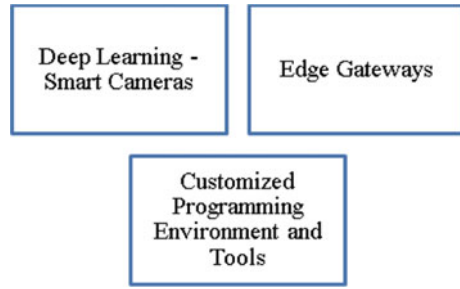
generated by the concerned industry; hence, new training is often the choice which makes the development expensive in time and cost. Transfer learning is the only available way out in such cases to save on cost and time.

Labelling Challenge: The labelling process in inspection requires experience in understanding the minute features that differentiate the good parts from defective ones. This often demands to be carried out by the experienced operator which is an additional time included in the regular production routine. Industries may not be in a position to spare that time.

Regulation Requirements: Most industries employing automated inspection are subjected to stringent regulatory and validation requirements particularly life sciences and pharmaceutical ones. This makes the industries to think twice before adopting deep learning since their outcomes are not always explainable.

Black Box: Since deep learning is a method that cannot be explained for its outcomes with arithmetic and logic, vision developers and end-users are generally less comfortable to ignore what is inside the black box.

3.2 Enablers of DL in Machine Vision

In the process of overcoming the many challenges deep learning is posed with when intended to be used for automated inspection, few enablers make the process easier. The key enablers for employing deep learning or automated inspection tasks are presented in Fig. 6.

These highly customized advances have accelerated the adoption rate of deep learning for automated inspection. The three key enablers are discussed in the subsequent section of hardware and software.

3.3 Deep Learning-Based Machine Vision Project Pipeline

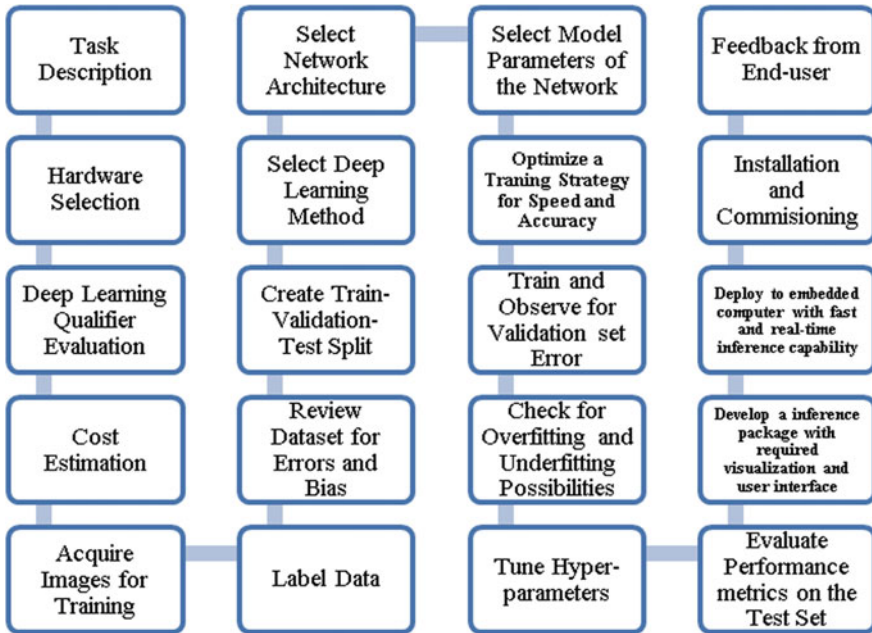A general deep learning project pipeline for an automated inspection task is presented in Fig. 7.

Fig. 7 Deep learning-based machine vision project pipeline

3.4 Deep Learning Hardware

Deep learning, unlike traditional machine vision applications, requires hardware with very specific attributes. The nature of hardware choice for deep learning required for training and inference is very different. Deep learning training is a highly compute-intensive task due to the following attributes of the training process.

- A large number of computing units (neurons) per layer
- A large number of layers of computing units
- A large number of iterations of forward-propagation, back-propagation and learning parameter update which involves lots of matrix operations
- Large spatial data. Numerous images and videos are generally the input, and they are very large in terms of the size of the memory compared to one-dimensional temporal data used in non-vision applications.

All the above mentioned factors demand large computation resources such as the memory of CPU, graphics processing unit like massively parallel computing hardware and vast GPU memory. Since training hardware is not commissioned in the industry, the remaining content of this section is dedicated to the hardware utilized for inferencing.

Deep learning inference is not as compute-intensive as the training process. The requirement is very different. Training time affects the time of deployment, not

productivity. Inference time directly affects productivity. This creates two fundamental requirements for deep learning inference hardware, namely, fast and real-time behaviour. Predictable outcomes and deterministic in time are the two main real-time behaviour. Though such attributes are achieved through a real-time operating system and the application code, the right choice of hardware to run such software is very important. There are various choices of inference hardwares which are listed below.

Embedded GPU-based Controllers: ARM-based micro-controllers with GPU cores embedded in a single integrated circuit are available in the form of a system-on-module (SOM) form factor. The Jetson platform from NVIDIA Corporation is an example of the same. These SOMs operate a real-time version of the Linux operating system with PC-like hardware interfaces. The inference code of the deep learning algorithm for automated inspection may be accelerated through the use of embedded GPU with special software tools that make the process extremely fast.

FPGA-based Controllers: Field programmable gate array is a fully configurable hardware which will have highly deterministic latency and perfectly real-time behaviour. They have other special benefits such as extremely low power consumption and heat generation. This category of hardware is rather new for deep learning applications. The usage of FPGA also gives the developer the benefit of the long production cycle, and since they do not have software implementation unlike other hardware choices, the deep learning network (such as a new type of convolutional neural network) as an intellectual property is protected.

Industrial Gateways: With the exponential growth in the adoption of deep learning in industries, computing hardware which is generally labelled as AI gateways have become very popular. They are somewhere between the fully-embedded controller and PC in terms of their hardware interfaces and computing capabilities. The AI gateway from Pleora and EDGE Ai Platforms from Adlink are typical examples of this category of products.

3.5 Deep Learning Software

After understanding the complexities in the computations involved in deep learning, it is natural to appreciate the importance of software tools used for training (development) and inference, as they are vital to the success of the application. Some of the important points to consider before selecting deep learning software tools are listed below.

- Compatibility with the operating system
- Cross-platform compilation capabilities, e.g. for embedded gateways.
- Compatibility with the hardware platform
- Support for common programming languages like C/C ++/Python
- Configurable tools for setting up the training process, e.g. train-validation-test split, vital statistics of the dataset, etc.
- Visualization tools for observing the training process, e.g. visualization atlases

- GPU acceleration capabilities and tools for converting the inference code to GPU-compatible form, e.g. NVIDIA TensorRT
- Logging and saving features, e.g. for weights, trained model and checkpoints
- Non-destructive overlays for visualizing the inspection results
- Transfer learning features
- Code reusability features.

There are many popular machine vision software companies that have their products enhanced with deep learning capabilities. Halcon and Merlic from Mvtec, Sherlock from Dalsa, Matrox Imaging Library from Matrox, Easysegment from Euresys are some of the popular platforms offering deep learning training and inference solutions supporting compatibility across multiple hardware platforms and operating systems.

4   Industries Using Deep Learning-Based Machine Vision

Automotive

Automotive being the major manufacturing units in most countries, a variety of inspection tasks are possible using deep learning. These include inspection of various subsystems such as tyre assembly, wheel fastener, airbag component, brake valve component, seat-belt component, airbag fabric, brake pad, cylinder, piston ring, transmission assembly, automotive metal stamping inspection, etc.

Medical Devices

Medical devices detection and classification of surfaces, detect errors on rolls of labels, detect scratches or stray particles on medical patches, inspect needlepoint quality, inspect the weld and adhesive placement, etc. One of the recent applications is the face mask quality inspection which involves detection of face mask components such as ear-bands, strap welds, presence of stains, rips, stitching errors, etc.

Pharmaceutical—Pills defects, Blister pack inspection

Deep learning in the pharmaceutical industry is widely used for quality control to detect defects in pills, inspect bottles and labels, in sorting of tablets, etc.

Consumer Products

The consumer product industry is one of the most diverse in terms of visual features of the products. Deep learning may be used for a variety of inspection tasks in consumer products, such as material quality inspection, label quality inspection, bottle cap inspection, knitting inspection, missing product detection, tamper and safety seal inspection, etc.

Food Industry

The food industry is one of the front-runners in using automated inspection for tasks such as contaminants inspection, physical form inspection and texture inspection.

Electronics

In electronics, deep learning may be used for integrated circuit lead cosmetic inspection, PCB inspection such as solder inspection, completeness check and connector pin inspection. Mobile devices such as smartphones and tablets have large LCD panel which have numerous possibilities of defects which are inspected by vision systems. These come under the category of surface inspection. Deep learning may further be used for cosmetic housing inspection and also for the analysis of cosmetic defects [17].

Agriculture

In agriculture, deep learning may be used for fruit classification, plant identification, leaf inspection, etc.

Logistics

Some of the potential applications of deep learning-aided machine vision applied to logistics include package classification, label identification, identification of empty rack spaces, etc.

5 Future Prospects

Some of the promising directions of changes and improvements that one can expect to see shortly in the context of deep learning's influence in the vision-based automated inspection include the following:

Cloud Tools

Cloud computing has influenced all sectors except the manufacturing sector wherein industrial vision applies. Though cloud storage and databases are utilized by large manufacturing companies, cloud computing is a new direction for implementing deep learning-based machine vision applications. There are software platforms, e.g. NeuralVision from Cyth [18], which can perform cloud-based computing for tagging, image handling statistical analysis, visualizations through simulation and solution generation. IP protection and security concerns were the primary reasons for industries not adopting cloud computing features. With large tech-giants like Google and Amazon offering top-notch cyber-security in their cloud computing platforms, there will be compelling reasons to use cloud resources for their computing involved in deep learning-based machine vision projects.

6   5G Technology

With 5G technologies around the corner, there is a serious expectation that the attention of industries might return to networks from microprocessors. This might severely influence the way deep learning algorithms are deployed and their potential in exploiting the network benefits.

Industry 4.0

The industry 4.0 revolution and deep learning happened parallelly for industries. After half a decade, there is a sense of convergence especially, since I4.0 adoption is influencing deep learning hardware and software with new types of data they are expected to broadcast and modern features of connectivity. This trend will continue to grow in the next five years with more IoT ready edge gateways offering the compute capability to run vision-based deep learning algorithms.

Quantum Computing

With the promise of quantum computing being tens to hundreds of times better than classical computing, deep learning-like compute-intensive tasks are likely to be the first few candidate applications to exploit its potential. The various stages of deep learning training can exploit the principles of quantum computing to generate ground-breaking time-performance.

Complex Algorithms

More and more complex algorithms in terms of the number of layers and the underlying principle are being introduced into industrial applications. This trend will continue to increase as more researchers have become industrial developers. This means the attitude of developers is very different from what it was for classical vision engineers. Such algorithms are expected to push the boundaries of the complexity of application that can be solved under the vision-based automated inspection framework.

Open Source Tools

Open source tools are the leaders in the deep learning software tool providers though custom proprietary software tools are available. It is currently being witnessed that the industries are willing to consider the open-source implementation of the vision-based deep learning segments of the larger application if not for the complete application.

The success of a company lies in its ability to recognize new technologies such as deep learning that creates scope for more automation and their readiness to spend on them. Such adoption of new technology does not essentially be a complete replacement to conventional technologies such as rule-based machine vision, rather they can be combined synergistically to result in new practices that will give the companies scale, efficiency, precision and financial growth for the next generation.

References

1. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In NIPS, pp. 1106–1114.
2. www.cognex.com, (visited on 25-10-2020).
3. www.mvtec.com, (visited on 22-10-2020).
4. Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations.*
5. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., & Rabinovich, A. (2015). Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition.*
6. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *IEEE Conference on Computer Vision and Pattern Recognition.*
7. Neuhauser, F. M., Bachmann, G., & Hora, P. (2020). Surface defect classification and detection on extruded aluminum profiles using convolutional neural networks. *The International Journal of Material, 13,* 591–603.
8. Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition.*
9. Girshick, R. (2015). Fast R-CNN. In *International Conference on Computer Vision.*
10. Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. In *Neural Information Processing Systems.*
11. He, K., Gkioxari, G., Piotr Doll´ar, & Girshick, R. (2017). Mask R-CNN. In *International Conference on Computer Vision.*
12. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., & Reed, S. (2015). SSD: Single shot multibox detector, arXiv preprint arXiv:1512.02325.
13. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *CVPR.*
14. Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. *Proceedings International Conference Medical Image Comput. Comput.-Assisted Intervention*, pp. 234–241.
15. Lin, T., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 936–944.
16. Chen, L., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2018). DeepLab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 40*(4), 834–848.
17. Yuan, Z. C., Zhang, Z. T., & Su, H. (2018). Vision-based defect detection for mobile phone cover glass using deep neural networks. *The International Journal of Precision Engineering, 19,* 801–810.
18. www.cyth.com, (visited on 22-10-2020).

Performance Improvement in Hot Rolling Process with Novel Neural Architectural Search

Srinivas Soumitri Miriyala, Itishree Mohanty, and Kishalay Mitra

Abstract State-of-the-art infrastructure, excellent computational facilities and ubiquitous connectivity across the industries have led to the generation of large amounts of heterogeneous process data. At the same time, the applicability of machine learning and artificial intelligence is witnessing a significant rise in academics and engineering, leading to the development of a large number of resources and tools. However, the number of research works and applications aimed at implementing data sciences to problems in process industries is far less. The proposed work aims to fill the niche by proposing Artificial Neural Network (ANN)-based surrogate construction using extremely nonlinear, static, high dimensional (32 features) noisy data sampled irregularly from inlet and outlet streams of hot rolling process in iron and steel making industry. Though ANNs are used extensively for modelling nonlinear data, literature survey has shown that their modelling is governed by heuristics thus making them inefficient for use in process industries. This aspect is of high relevance in contemporary times as hyper-parameter optimization, automated machine learning and neural architecture search (NAS) constitute a major share of current research in data sciences. We propose a novel multi-objective evolutionary NAS algorithm to optimally design multi-layered feed-forward ANNs by balancing the aspects of parsimony and accuracy. The integer nonlinear programming problem of ANN design is solved using binary coded Non-Dominated Sorting Genetic Algorithm (NSGA-II). ANNs designed for the hot rolling process are found to demonstrate an accuracy of 0.98 (averaged on three outputs) measured in terms of correlation coefficient R^2 on the test set. The successful construction of accurate and optimal ANNs provides a first-of-its-kind model for the hot rolling process in the iron and steel making

S. S. Miriyala · K. Mitra (✉)
Global Optimization and Knowledge Unearthing Laboratory, Department of Chemical Engineering, Indian Institute of Technology Hyderabad, Sangareddy 502285, Telangana, India
e-mail: kishalay@che.iith.ac.in

S. S. Miriyala
e-mail: ch13m15p000002@iith.ac.in

I. Mohanty
Research & Development, Tata Steel Limited, Jamshedpur 831001, Jharkhand, India
e-mail: iti.mohanty@tatasteel.com

industry. The proposed method can minimize the chances of over-fitting in ANNs and provides a generic method applicable to any kind of data/model from process industries.

Keywords Micro-alloying · Artificial neural networks · Neural architecture search · Evolutionary algorithms · Multi-objective optimization · INLP formulation · Hot rolling process · Surrogate modelling · Hyper-parameter optimization · NSGA-II

1   Introduction

Micro-alloying is a significant process in iron and steel making industries, which results in remarkable changes in the mechanical properties of the steel [1]. In recent times, this process has gained immense applicability, since through this process, it is possible to improve the strength of steel without compromising on its quality [1]. One important measure of quality during the hot rolling of micro-alloyed steels is the mechanical property, as optimization and control of mechanical properties govern the applicability of the end product. The mechanical properties of micro-alloyed steels are determined traditionally through destructive testing. This procedure is laborious and cost-intensive. While modelling the mechanical properties is an alternative, however, with micro-alloying in place, it becomes extremely challenging to model the mechanical properties using the first principles. Without the models in place, it is nearly impossible to conduct optimization and control studies at the industrial level due to the iterative nature of these algorithms involving repetitive function evaluations [2]. Thus, a suitable model (preferably not relying on first principles) mapping the operating conditions, chemical compositions and other design parameters of the hot rolling process with the mechanical properties of micro-alloyed steels is the need-of-the-hour. However, at the same time, sophisticated instrumentation, ubiquitous connectivity and automation have led to the generation of vast amounts of heterogeneous process data, which allow for the construction of fast and accurate learning algorithms to emulate high dimensional nonlinear process data, such as that arising from the hot rolling process in iron and steel making industry. Therefore, a possible solution to overcome the aforementioned issues is to streamline the process with Data Science (DS) [3].

Data science is broadly defined as (a) management, (b) Machine Learning (ML) and (c) visualization of data, among which machine learning lies at its core [4]. The first task deals with organizing, storing, accessing and sharing the data. The second task employs mathematical algorithms for capturing the hidden trend in data, for example, developing prediction models, determining the underlying structure of the data and developing experience-based models. The third task consists of methods that can present the insights of the data and in turn help the organizations/industries for making important decisions based on data analysis. Owing to the numerous advantages that the DS-based tools provide, the applications of DS are widespread

in various domains of research. Some of them include supply chain and vehicle route planning [5], natural language processing [6], image recognition [7], climate change [8], agriculture [9], healthcare [10], drug design [11], gaming [12], stock market prediction [13] and online marketing [14]. Not to be left out, the field of chemical and process engineering is also witnessing substantial growth due to technological advances in experimentation, thereby leading to the generation of big data [4, 15, 16]. Recently, the chemical engineering community has seen benefits from DS such as data-driven modelling [17], intelligent supervisory control [18], energy systems management [19], fault analysis [20], building better models in understanding catalysis research [21], crystal identification and discovery [22], surrogate modelling and optimization [23], uncertainty handling in chemical processes [24], molecular modelling and simulation [25] and so on. When it comes to the area of process optimization and control, a popular application of machine learning is the development of data-driven surrogate models to emulate the high-fidelity time expensive physics- based model. Once trained to the desired accuracy, these surrogates work by obscuring the time-expensive physics-based models from the optimizer, thereby accelerating the arduous process of optimization and control. They are known to work with a wide variety of optimization algorithms and processes as can be seen in [26].

One classic example of surrogate models is the Artificial Neural Networks (ANN), capable of modelling the nonlinear behaviour of data/complicated models [27]. ANNs are advantageous over several other prominent classes of surrogates or data-based models from the paradigm of machine learning. However, issues associated with their design and modelling, particularly those dealing with heuristics driven determination of architecture, activation function and sample size, severely degrade their capability as surrogate models [2]. A significant amount of research is carried out to alleviate ANNs with these serious modelling issues. For example, [28] presented a framework based on Mixed Integer Nonlinear Programming Problem (MINLP) to design ANNs [28, 29] tried a weighted-sum approach based on meta-heuristics [29], Biothias et al. (2012) presented a multi-objective optimization framework [30, 31] came up with a novel method based on a sequential sampling strategy for determining the optimal training sample size of ANNs [31]. Other prominent works in the field of hyper-parameter optimization can be found in [32–38]. The recent trend showed the shift towards the application of Bayesian optimization and reinforcement learning for the determination of hyper-parameters (mainly the architecture design) in neural networks [39, 40].

In this work, we propose a novel multi-objective neural architecture search strategy to overcome the issues associated with feed-forward neural networks for modelling high dimensional (32 features) data obtained from the hot rolling process in the iron and steel making industry. Unlike any other method proposed in the literature, the current work focusses on the design of ANNs and the determination of optimal activation function, while balancing the aspects of over-fitting and accuracy. The proposed algorithm is not only fast, leading to its real-time implementation, but also generic to be applied to any kind of data from industry without any constraints on input–output dimensions and extent of nonlinearity. Since the data is obtained directly from the

plant, it is first pre-processed to minimize the measurement noise and retain only the relevant information of the process. The resultant data is then used to run the proposed algorithm capable of optimally designing the ANNs. In the current work, the inputs to the ANN model constitute 29 parameters involving the chemical compositions, physical properties of steel fed to the rolling mill and operating conditions of the rolling mill. The desired mechanical properties in the final product such as high tensile strength, yield and elongation constitute the 3 outputs of the ANN model. The obtained optimal ANNs were able to emulate the high dimensional nonlinear data with an accuracy of 98% on the test set (unseen data). While the algorithm for ANN design serves as the major novelty, a detailed description of multi-layered ANN construction, multi-objective nature of neural architecture search strategy and application of the proposed algorithm to real, nonlinear, industrial, high dimensional data for constructing a data-driven model to enable data-driven optimization and control of rolling mill are other highlights of the proposed work. In the rest of the article, the details of the proposed algorithm and process description of hot rolling in steel making industries are presented in the 'Formulation' section, followed by the 'Results' of ANN design and model building before summarizing the work in the 'Conclusions' section.

2 Formulation

2.1 Evolutionary Neural Architecture Search for Optimal ANN Design

ANNs are graphical models which were designed to mimic the functioning of the human brain to perform the tasks of classification and regression [41]. The basic functioning unit of the nervous system called the nerve cell is modelled using a node comprising the summation unit and the activation function to mimic the synapsis and nucleus, respectively, as shown in Fig. 1. The summation unit would result in a weighted sum of inputs to the node, while the activation function is responsible for
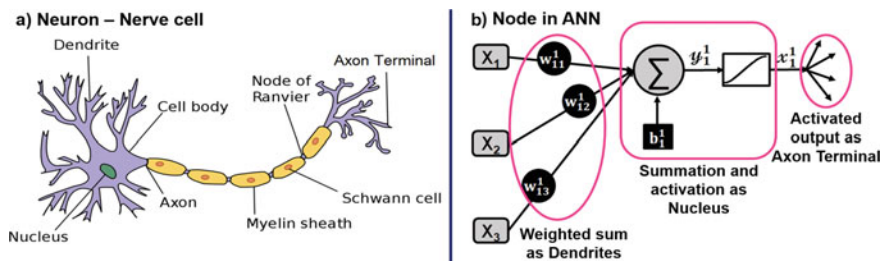

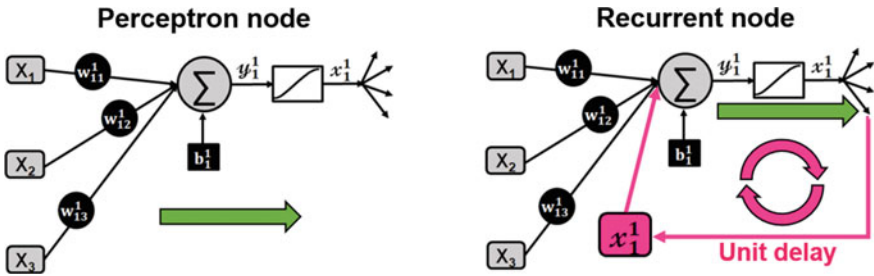**Fig. 1** Comparison between Neuron and Node

Fig. 2 Comparison between feed-forward (perceptron) and feedback (recurrent) node

nonlinear transformations of the data. ANNs are a collection of several such nodes and are broadly classified into feed-forward and recurrent neural networks based on the feed-forward and feedback connections, respectively, between the nodes in the network as shown in Fig. 2. While feed-forward or multi-layered perceptron networks are used to model static data, recurrent networks are designed specifically to emulate dynamic datasets. A third category called convolutional neural networks is also defined in literature to model image-based datasets. In this work, we primarily deal with static data, and hence, only multi-layered perceptron networks are discussed in the rest of the manuscript. Readers interested in RNNs and CNNs can refer [42, 43], respectively, for more details.

The set of inputs constitute the input layer from which the information is passed on to a series of hidden layers, finally culminating into the output layer. A hidden layer is a collection of several nodes functioning in parallel. In a multi-layered perceptron network, the information flows in only a forward direction to generate the outputs of the ANN. The network generated outputs are compared with the original outputs or labels (in case of classification) to quantify the amount of deviation. This measure called the loss function is then minimized by adjusting the trainable parameters in the network: the weights and biases as shown in Fig. 3.

The necessity of outputs or labels for every set of inputs categorizes ANN modelling as supervised machine learning. Further, the graphical structure of the ANNs allows the analytical determination of gradients of the loss function with respect to weights through the procedure called backpropagation. Thus, classical gradient-based optimization algorithms such as steepest descent [41] or the Newtonian techniques [41] are employed to train the ANNs.

Performing only weight training in ANNs might often lead to a situation where they no longer remain capable of interpolation or prediction for unseen data. This phenomenon called over-fitting (to the training data) is generally avoided in ANNs by dividing the given data into three parts: the first portion called training set is used to train the ANN model, the second portion called validation set is used to check over-fitting during training and the third portion called test set is used to check the performance of trained ANN model on unseen data. The procedure to prevent over-fitting in ANNs using validation set is called early-stopping, where, the training exercise is terminated when it is observed that further training does not result in
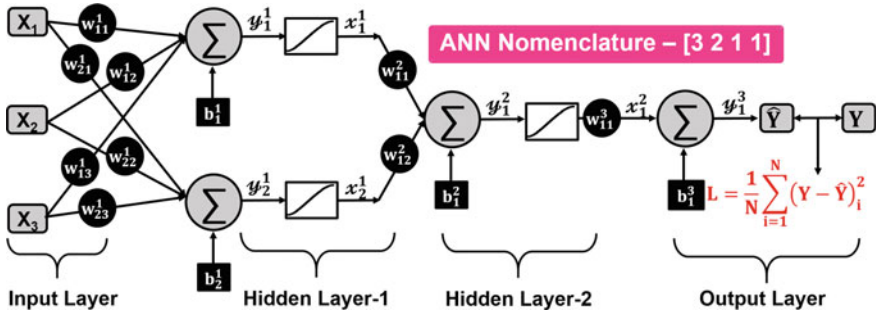
Fig. 3 A multi-layered perceptron network with 3 inputs, 2 hidden layers and 1 output. During forward pass, information flows in forward direction (left to right) to generate a Mean Square Error (MSE) loss L over all N training points, and during backpropagation, the information flows from right to left

improvement of validation loss or typically leads to an increase in validation loss, thereby suggesting over-fitting to the training set. While other techniques like regularization [41] also exist in literature to prevent over-fitting, early-stopping is still an easy and effective method to ensure generalization of ANNs. The presence of a large number of parameters (weights and biases), an ensemble of nonlinear activation functions, effective training method through backpropagation and techniques to prevent over-fitting make ANNs one of the best ML techniques to model nonlinear high dimensional datasets.

Apart from the usual parameters in ANNs, the number of hidden layers, number of nodes and activation function are also needed to be determined a priori. Collectively called the hyper-parameters, their estimation, as being realized in a recent literature survey, has a significant role in governing the efficiency of ANN models. Being integral in nature, the hyper-parameter estimation when combined with weight training exercise leads to the formulation of the MINLP problem, which is NP-hard to solve. This problem of NAS is therefore, taken up in different ways, among which the formulation involving reinforcement learning [39] and Bayesian optimization [40] have been most successful. In this work, we implement an evolutionary NAS strategy to design the optimal ANNs using a multi-objective optimization formulation. In the proposed method, the MINLP formulation is divided into two levels where the outer level deals with hyper-parameter estimation and the inner level with the conventional weight training exercise. This makes the hyper-parameter estimation exercise an INLP formulation and separates it from the NLP formulation of weight training, but at the same time, since the two levels of optimization are solved in tandem, it also ensures a simultaneous determination of the hyper-parameters and parameters of ANN, thus making it parameter-free. While methods like early-stopping and regularization prevent over-fitting in the inner loop, the proposed formulation further minimizes the chances of over-fitting in the outer loop optimization. This is enabled by considering the variance-bias trade-off in ML models: the biasness to consider a simpler ML model (with less number of parameters) may often lead to large variance

between the predicted results and original values. Thus, we formulated the objectives as minimizing the number of parameters in ANN (which are correlated with the architecture of ANN) and maximizing the test set accuracy to balance the aspects of parsimony and over-fitting. The architecture (number of hidden layers and number of nodes in each hidden layer) and activation function type serve as the decision variables in the proposed formulation for hyper-parameter optimization as shown in Eq. 1.

$$\min_{n^i \text{ and } n^{TF}} \left(-R_{test}^2 \text{ and P} \right) \tag{1}$$

$$\text{such that, } L \leq n^i \leq \eta_i \text{ where } L = \left\{ \begin{matrix} 1, \text{ if } i = 1 \\ 0, \text{ if } i > 1 \end{matrix} \right\}$$

$$\forall i = 1, 2 \ldots \lambda; \, n^i, \eta_i \text{ and } \lambda \in \mathbb{Z}_+ \text{ and } n^{TF} \in \{1, 2\}$$

$$\text{where, } R_{test}^2 \text{ is the measure for accuracy on test set} = \left(\frac{\text{cov}\left(\mathbf{Y}, \hat{\mathbf{Y}} \right)}{\sqrt{\text{var}(\mathbf{Y}) \text{var}\left(\hat{\mathbf{Y}} \right)}} \right)^2$$

$$\text{cov}\left(\mathbf{Y}, \hat{\mathbf{Y}} \right) = N_T \sum_{i=0}^{N_T} \mathbf{Y}_i \hat{\mathbf{Y}}_i - \sum_{i=0}^{N_T} \hat{\mathbf{Y}}_i \sum_{i=0}^{N_T} \mathbf{Y}_i$$

$$\text{var}(\mathbf{Y}) = N_T \sum_{i=0}^{N_T} \mathbf{Y}_i^2 - \left(\sum_{i=0}^{N_T} \mathbf{Y}_i \right)^2$$

$\mathbf{Y}$ is the original outputs, $\hat{\mathbf{Y}}$ is the predicted outputs and N_T is number of points in the test set. P is total number of parameters in the ANN model. n^i is number of nodes in hidden layer i and n^{TF} is the parameter for choice of activation function (1 for tan-sigmoidal and 2 for log-sigmoidal), while η_i and λ are upper bounds on n^i and i, respectively, and $\mathbb{Z}_+$ is set of positive integers. Multi-objective nature of proposed formulation, INLP formulation and lack of gradient evaluation provided scope for implementation of population-based evolutionary optimization algorithms to solve the NAS problem. In this work, since we have two conflicting objectives and integral decision variables, we implemented binary coded Non-Dominated Sorting Genetic Algorithm (NSGA-II) [44]. The flow of the algorithm is shown in Fig. 4.

We start by initializing the number of generations (N_{gen}) and the population size (N_{pop}) in NSGA-II. While any values for the upper bounds η_i and λ (number of nodes in each hidden layer and number of hidden layers, respectively) can be used, in this work, we have set $\lambda = 3$ and $\eta_1 = 8, \eta_2 = 8$ and $\eta_3 = 7$. Since binary NSGA-II is used, each decision variable is represented by a binary sub-string composed of 0's and 1's. To ensure integral values between the lower bounds (as shown in Eq. 1) and the aforementioned upper bounds on decision variables, n^1, n^2 and n^3 are
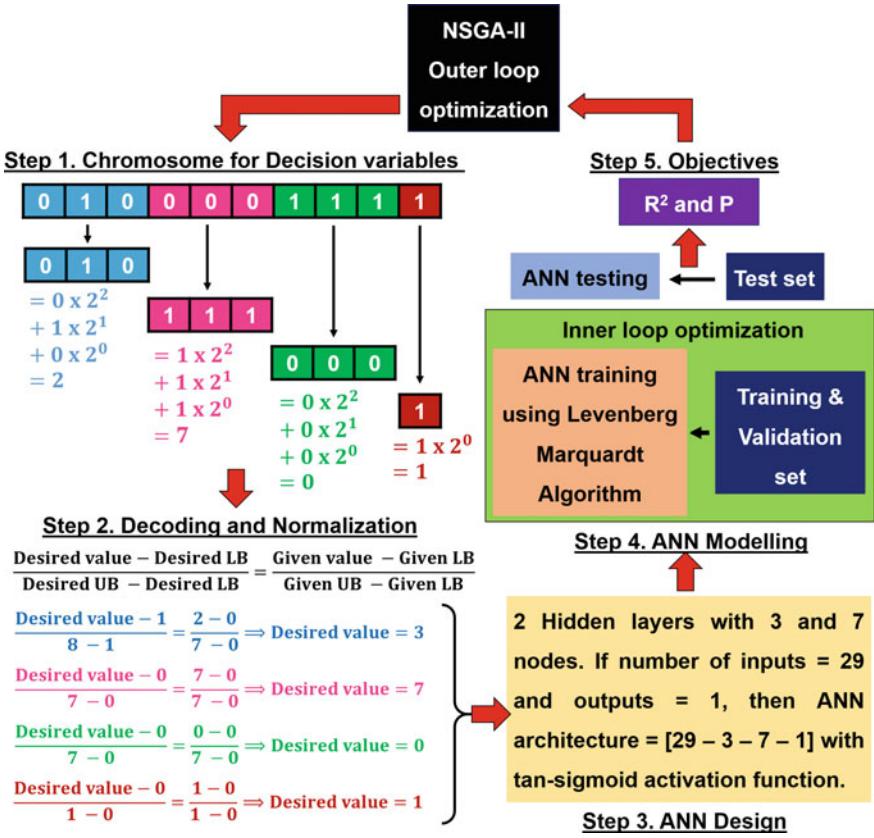
Fig. 4 Flowsheet of proposed algorithm for evolutionary NAS using NSGA-II

represented with 3 binary sub-strings of 3 bits length for enabling them to create 8 discrete integral values (n^1 will be an integer between 1 to 8, n^2 will be an integer between 0 to 7 and n^3 will be an integer between 0 to 7), and n^{TF} is represented with a binary string of length 1 to enable binary output. Concatenating these 4 binary strings horizontally, a single 10-bit long candidate called chromosome is created, where each bit is called a gene. N_{pop}, such chromosomes constitute the population in one generation of NSGA-II. Thus, for a given chromosome, architecture and activation choice are decoded to create the ANN as demonstrated with an example in Fig. 4. Once the ANN configuration is obtained, the inner loop optimization is performed to train the weights and test the trained network with a test set and evaluate the corresponding R^2_{test} as per Eq. 1. Similarly, the second objective, P can also be evaluated for the given architecture by counting the weights and biases in the network. This exercise is repeated for all N_{pop} candidates to complete one generation of NSGA-II. Upon successful evaluation of one generation, all the unique solutions are kept in a database to avoid redundant calculations in future generations of NSGA-II.

Selection, crossover, mutation and elitism operations [44] are performed to create a subsequent generation of populations. This procedure is repeated till convergence of the NSGA-II algorithm. Finally, the rank-1 solutions are considered as the final Pareto front in which each point correlates with an ANN architecture. A suitable higher order information such as K-fold cross validation [45] or Akaike Information Criterion [46] or any other model evaluation criterion [44] can be used to select one architecture from the list which serves as the final optimally designed ANN solution capable of maximum accuracy and minimal over-fitting.

2.2 Hot Rolling Process Description and Data Generation

Prediction of mechanical properties of steel has been a subject of research for the last two decades. Traditionally, the desired mechanical properties are achieved using process metallurgy knowledge and industry experience. Mechanical properties of steel depend not only on the inherent chemistry but also the microstructure that gets evolved during the manufacturing process [47]. In micro-alloy steel manufacturing, the evolution of microstructure is very complex and dynamic, making the predictions very less consistent. This in turn increases the amount of effort and time required to produce new steel with desired properties. With the advances in physical metallurgy, thermo-mechanical processing and increasing use of artificial intelligence, attempts have been made to develop models that can increase the consistency in prediction of mechanical properties of micro-alloyed steels. As a result, several models have been developed by various research groups in universities, steel research labs [48–50] and there have been attempts to put these models into applications in production of steel products. These models mostly concentrate on hot rolling where the processing conditions are closely controlled in order to achieve the final properties.

In the Thin Slab Casting and Rolling (TSCR) process or more commonly known as Compact Strip Process (CSP) as shown in Fig. 5, first, molten steel is cast into slabs
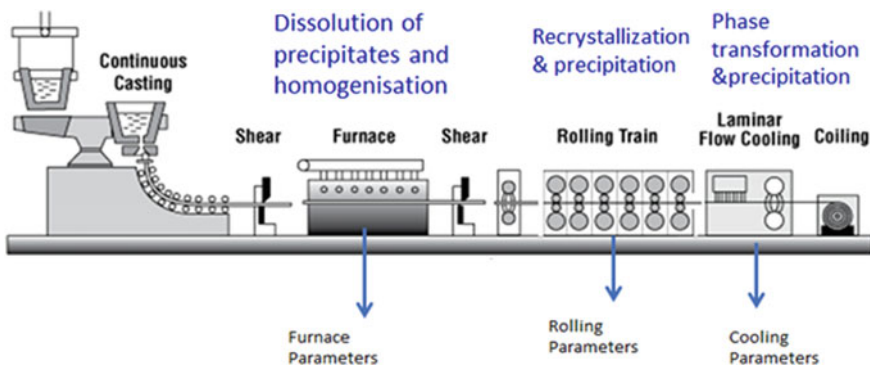

Fig. 5 Schematic of compact strip process

of thickness of ~50 to 70 mm using a thin slab continuous caster. The cast steel is then sheared and passed through an equalizing furnace at 1150 °C where it stays for around 25 min. It then enters the finishing mill. Because of the absence of the roughing mill in this process, the grains in the slab at the entry to the first stand of the finishing mill are very large (average grain size ~600 µm). Also, the microstructure of thin slab at this position will have cast microstructure. Appropriate thermo-mechanical processing is required to be imparted in the finishing mill in order to break this cast microstructure and reduce the austenite grain size. The slab is gradually rolled as it passes through the stands of finishing mill. The steel is then cooled using the laminar cooling process and coiled at room temperature. Input data comprising the operating conditions, chemical compositions and design parameters are collected directly from the inlet streams. The mechanical properties for the corresponding data are obtained from laboratory study and analysis of samples collected from outlet streams. The process data is described in Table 1.

3 Results and Discussions

3.1 Data Pre-processing

The considered process contains 29 inputs and 3 outputs as shown in Table 1. The data for training the neural networks in this study is sampled at irregular intervals from the inlet and outlet streams in the industry. Due to errors in instrumentation, it is safe to assume that the chances of data being corrupted with measurement noise are relatively high. To check and confirm whether the data (particularly the outputs) is prone to noise, we passed it through a low-frequency filter, such as moving average. Generally, these filters allow only the data having frequency less than a pre-fixed value to pass through, however, they do not confirm that the high frequency component is noise. Therefore, the resultant high frequency component is analysed using an auto-correlation plot. If the analysis shows that there is no statistically significant correlation in the data, then it can be hypothesized that the high frequency component is noise. Since the extent of noise to be filtered is not known a priori, certain amount of hit-and-trial exercise has to be performed to approximately determine the amount of noise to be filtered out from the data. This entire analysis was performed in this work for every output feature.

Figure 6 presents the auto-correlation plots of high (noise) and low frequency (data) components for all three outputs. The presence of data only between the 99% and 95% confidence lines indicates that the data is white noise. This characteristic feature can be seen in subfigures Fig. 6a–c, while it is significantly different from the features observed in subfigures Fig. 6d–f.

To confirm that the data used for plotting the subfigures Fig. 6a–c is white noise, we plotted the histograms as shown in Fig. 7. These histograms (Gaussian shape) confirm that the data is indeed white (measurement) noise.

Table 1 Description of inputs and outputs considered in the proposed work

SL. No.	Inputs of ANN model		Outputs of ANN model
1	Slab dimensions	Slab thickness	Yield strength
2		Slab width	Ultimate tensile strength
3		Slab length	Elongation
4	Compositions	Aluminium	
5		Boron	
6		Carbon	
7		Chromium	
8		Copper	
9		Manganese	
10		Nitrogen	
11		Niobium	
12		Phosphorous	
12		Sulphur	
14		Silicon	
15		Titanium	
16		Vanadium	
17	Operating conditions	Entry temperature	
18		Exit temperature	
19		Duration	
20		Finish rolling	
21		Coiling temperature after rolling	
22		Actual roll 1 gap	
23		Actual roll 2 gap	
24		Actual roll 3 gap	
25		Actual roll 4 gap	
26		Actual roll 5 gap	
27		Actual roll 6 gap	
28		Roll 6 speed	
29		Slab thickness	
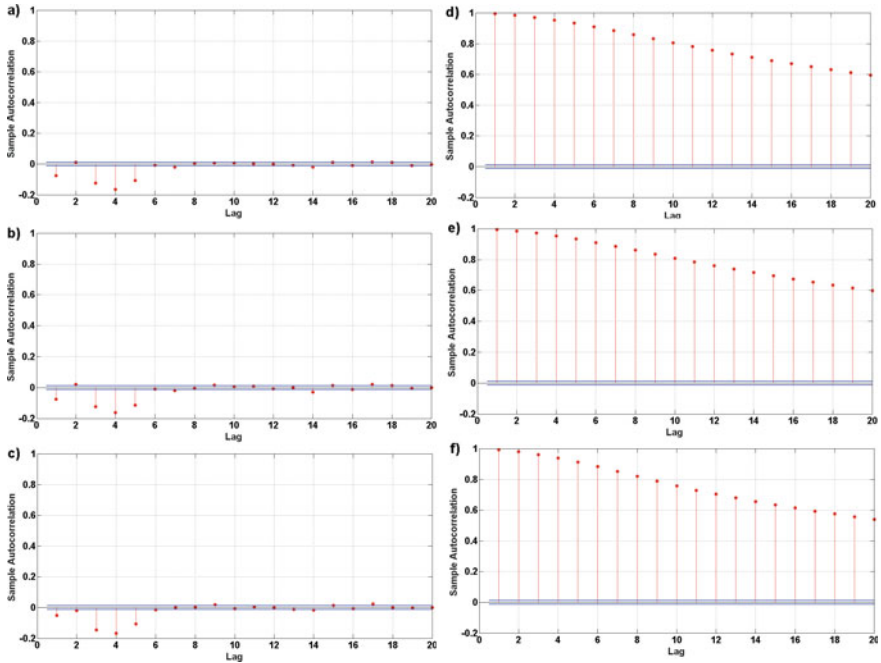
Fig. 6 Auto-correlation plots. Subfigures a, b and c are for high frequency components and d, e and f are for low frequency components for output 1, 2 and 3, respectively

On the other hand, the data in subfigures Fig. 6d–f demonstrates significant level of correlation. Also, the scatter plot of low-frequency component for all three outputs in Fig. 8 indicates varying non-zero mean, thereby allowing us to hypothesize that statistically significant amount of white noise is filtered from the measured data, and the remaining component is true characteristic of the process. Figure 8 also demonstrates the extent of nonlinearity in the data.

After the pre-processing of the data and filtering out the noise, the remaining component, indicative of process characteristics, is used for ANN modelling. Since the process is running at steady state, it is assumed that the dynamics in the data do not change with time (static nonlinear data) allowing us to model it using feed-forward or the multi-layered perceptron networks.

3.2 Neural Architecture Search

The proposed multi-objective optimization algorithm is simulated to build the ANN models. Since we have 3 outputs in the process, 3 such simulations were run to build 3 Multiple Input Single Output (MISO) ANN models as opposed to a single Multiple Input Multiple Output (MIMO) ANN. Since the data is static, outputs are
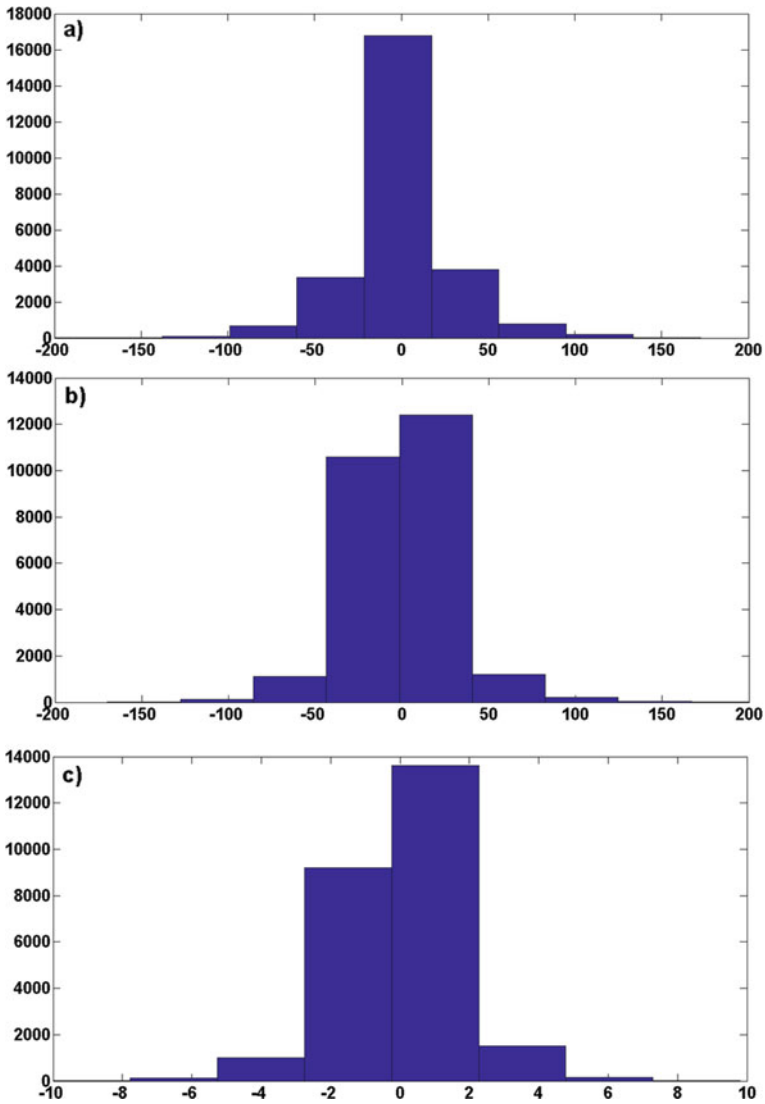
Fig. 7 Subfigures a, b and c present the histogram of high frequency components in the data for outputs 1, 2 and 3, respectively

uncorrelated, therefore resulting in no difference between the predictability of MISO and MIMO ANN models. However, constructing 3 MISO models allowed us to reduce the complexity in ANN training and also improve the speed of the proposed evolutionary NAS algorithm. Thus, 3 MISO ANN models were built in this work. As described previously, the outer loop was solved using NSGA-II, whose credentials
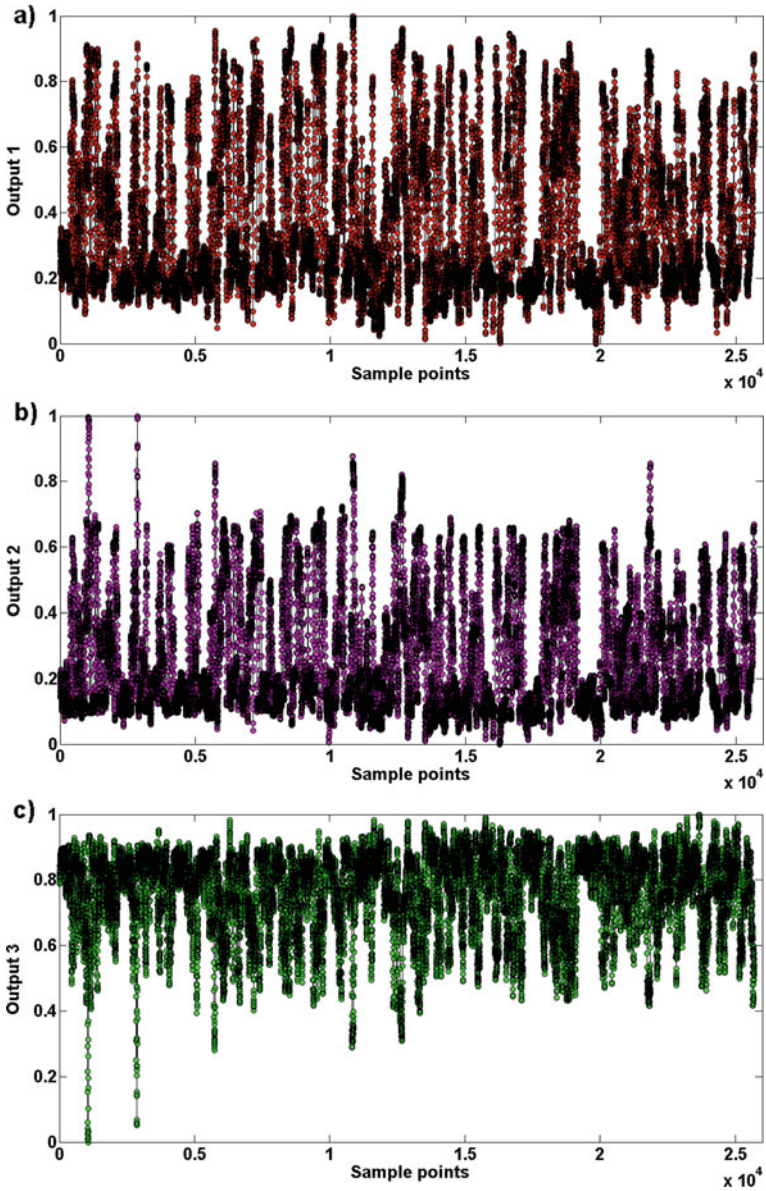
Fig. 8 Subfigures a, b and c present the scatter plots of low frequency components in the data for outputs 1, 2 and 3, respectively

Table 2 Parameter setting in NSGA-II

| SL. No.	Parameters	Values
1	Binary variables	4
2	Real variables	0
3	Generations	100
4	Populations	100
5	Probability of crossover	0.9
6	Crossover type	Uniform
7	Probability of mutation	0.01
8	Initialization	Random
9	Upper and lower bounds on decision variables	[8 7 7 2] & [1 0 0 1]

are presented in Table 2. The inner loop optimization (ANN weight training) was solved using Levenberg Marquardt Algorithm (LMA) [41].

Since the proposed NAS strategy is multi-objective in nature with 2 objective functions, 2-dimensional Pareto fronts were obtained as shown in Fig. 9, where each point is an optimal ANN architecture with optimal activation function. The convergence of Pareto front in each case was ensured by starting the NSGA-II algorithm with different initial random populations and also running NSGA-II to significantly large number of generations (~1000). No change in final Pareto confirmed the convergence of proposed NAS algorithm.

A single ANN architecture needs to be selected from the list of Pareto solutions for the purpose of utilizing it as a surrogate for the hot rolling process. A suitable Higher Order Information (HoI) needs to be applied to select a single solution [44]. In this work, we chose to implement AIC as HoI: ANN model with least AIC value is selected. This is because, AIC is a unique model selection criterion, which penalizes the models (ANNs in this case) when they have large number of parameters (weights and biases in this case). By doing this, it essentially finds the ANN model which is least over-fitted. Other similar state-of-the-art model evaluation criterion in literature is K-fold cross validation [46]. However, its computationally intensive procedure allowed us to prefer AIC. The list of Pareto solutions along with AIC values are shown in Tables 3, 4 and 5. Selected architectures along with their performance with respect to test set are shown in Table 6.

3.3 Discussions and Future Scope

The evolution of multi-layered perceptron networks in this study signify the need for a proper design strategy capable of disqualifying the popular heuristic belief of considering only single hidden layered networks. Further, it can be observed that, the algorithm was able to find architectures with more than one hidden layer but
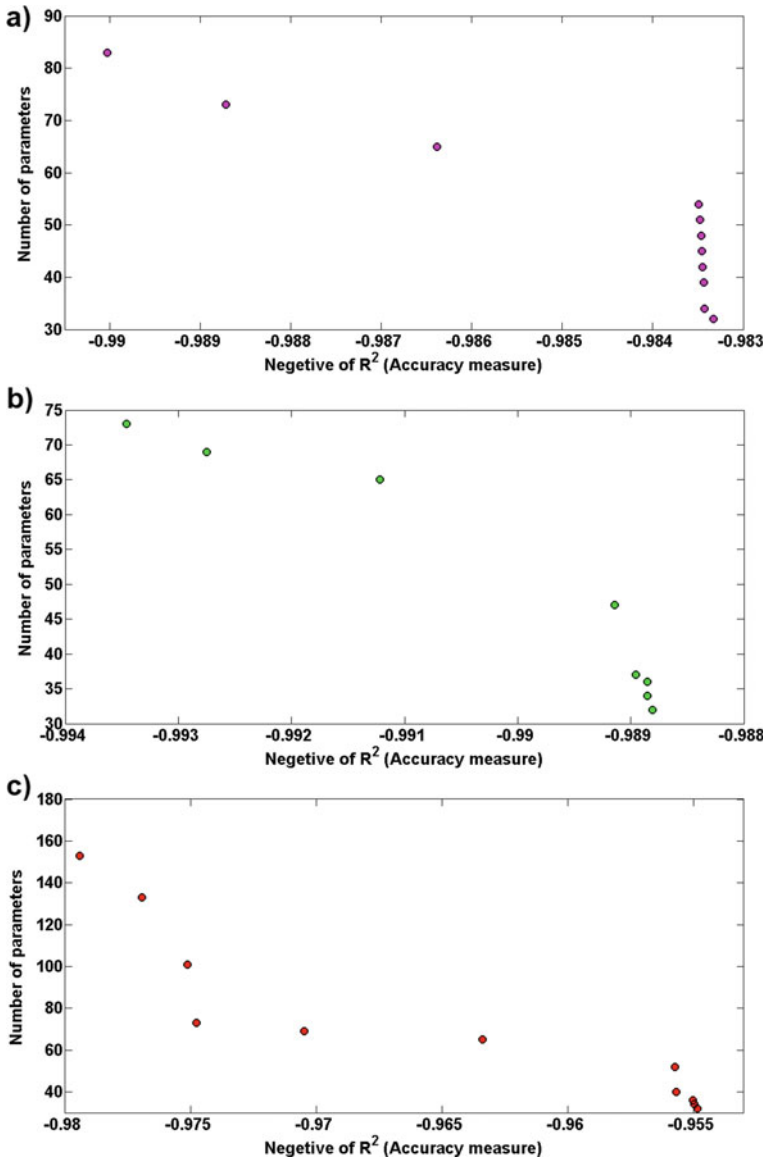
Fig. 9 Subfigures a, b and c present the Pareto fronts obtained by solving the proposed NAS algorithm to build ANNs for emulating outputs 1, 2 and 3, respectively

Table 3 List of Pareto solutions obtained by solving the proposed NAS algorithm for output 1 along with corresponding AIC values. Entry in bold indicate the selected architecture

SL. No.	n^1	n^2	n^3	n^{TF}	R^2_{test}	P	RMSE	AIC
1	1	0	0	1	0.98333	32	0.051306	−59334.8
2	1	1	0	1	0.98343	34	0.051153	−59390.6
3	1	2	1	1	0.98344	39	0.051137	−59387
4	1	3	1	1	0.98345	42	0.05112	−59387.8
5	1	4	1	1	0.98346	45	0.051101	−59388.9
6	1	5	1	1	0.98347	48	0.051092	−59386.7
7	1	6	1	1	0.98348	51	0.051067	−59390.2
8	1	7	1	1	0.98349	54	0.051055	−59389.2
9	2	1	0	1	0.98639	65	0.046362	−61295.5
10	2	3	0	1	0.98872	73	0.042199	−63161.3
11	**2**	**5**	**1**	**1**	**0.99003**	**83**	**0.03967**	**−64377.3**

Table 4 List of Pareto solutions obtained by solving the proposed NAS algorithm for output 2 along with corresponding AIC values. Entry in bold indicate the selected architecture

SL. No.	n^1	n^2	n^3	n^{TF}	R^2_{test}	P	RMSE	AIC
1	1	0	0	1	0.98881	32	0.041255	−63695.7
2	1	1	0	1	0.98885	34	0.041173	−63731.5
3	1	1	1	1	0.98886	36	0.041168	−63730
4	1	2	0	1	0.98896	37	0.040979	−63819.8
5	1	2	3	1	0.98914	47	0.040628	−63971.8
6	2	1	0	1	0.99122	65	0.036576	−66037.5
7	2	2	0	1	0.99275	69	0.03321	−67960.1
8	**2**	**3**	**0**	**1**	**0.99346**	**73**	**0.03153**	**−68990.1**

very few nodes in each layer. Once again this ability disapproves another popular heuristic which necessitates the presence of a large number of nodes in hidden layer. Often such model might work well with training data but it fails to generalize due to the outburst of parameters. The proposed algorithm precisely prevented this aspect by adding the conflicting objective of minimizing total nodes while maximizing the accuracy.

Therefore, this formulation not only prevented the emergence of over-fitted models but also allowed for ANN models with smaller configurations which hold immense importance when implementing them online as they require significantly less amount of memory and processing time compared to ANNs with large configurations. However, this feature of proposed NAS strategy does not demean the significance of deep neural networks. Instead, it only demonstrates that the best judge to decide the configuration of the ANN is the data itself. Since any number of layers and

Table 5 List of Pareto solutions obtained by solving the proposed NAS algorithm for output 3 along with corresponding AIC values. Entry in bold indicate the selected architecture

SL. No.	n^1	n^2	n^3	n^{TF}	R^2_{test}	P	RMSE	AIC
1	1	0	0	1	0.95485	32	0.084321	−49398.4
2	1	1	0	1	0.95498	34	0.084189	−49425.8
3	1	1	1	1	0.95503	36	0.084144	−49432.5
4	1	3	0	1	0.95569	40	0.08353	−49571.1
5	1	3	3	1	0.95575	52	0.083464	−49562.7
6	2	1	0	1	0.96341	65	0.075968	−51418.8
7	2	2	0	1	0.9705	69	0.06819	−53571.2
8	2	3	0	1	0.97478	73	0.063036	−55134.9
9	3	2	0	1	0.97513	101	0.062643	−55204.2
10	3	3	6	1	0.97695	133	0.060294	−55904.3
11	**3**	**5**	**6**	**1**	**0.97943**	**153**	**0.057064**	**−56965.6**

Table 6 Optimally designed ANN models for emulating the hot rolling process, obtained using the proposed evolutionary NAS strategy

Output. No.	n^1	n^2	n^3	n^{TF}	R^2_{test}	P	RMSE	Sample size for training, validation and testing
1	**2**	**5**	**1**	**1**	**0.99003**	**83**	**0.03967**	**10000, 5000, 10000**
2	**2**	**3**	**0**	**1**	**0.99346**	**73**	**0.03153**	**10000, 5000, 10000**
3	**3**	**5**	**6**	**1**	**0.97943**	**153**	**0.057064**	**10000, 5000, 10000**

nodes can be explored using the proposed algorithm, we believe it will work equally well for deep neural networks provided enough computational time is allowed. Similarly, given enough computation resource, the proposed method can also be used for optimally designing Recurrent Neural Networks and Deep-RNNs demonstrating the generic abilities of novel algorithm. At the same time, the speed of the proposed evolutionary NAS strategy can be improved by several folds on GPUs as it has immense scope of parallelization. These aspects lay the future scope of proposed work.

4 Conclusions

The applicability of ANNs to build surrogate models for high dimensional nonlinear heterogeneous process data from iron and steel making industry is explored in the current work. It is realized that heuristics related to modelling ANNs play a significant role in their accuracy, and therefore, an effort is made to eliminate the heuristics while designing ANNs. In this work, we proposed a novel algorithm for constructing ANNs

by determining optimal number of hidden layers and nodes in each layer along with optimal activation function choice. The proposed methodology was able to build multi-layered ANNs capable of emulating the process data with 98% accuracy. While the method was shown for constructing feed-forward networks, it can be easily scaled up to deep neural networks and recurrent neural networks, thus making it generic and capable of working with data from any domain.

References

1. Mittal, P., Mohanty, I., Malik, A., & Mitra, K. (2020). Many-objective optimization of hot-rolling process of steel: A hybrid approach. *Materials and Manufacturing Processes, 35*(6), 668–676.
2. Miriyala, S. S., Mittal, P., Majumdar, S., & Mitra, K. (2016). Comparative study of surrogate approaches while optimizing computationally expensive reaction networks. *Chemical Engineering Science, 140,* 44–61.
3. Van Der Aalst, W. (2016). Data science in action. In *Process mining* (pp. 3–23). Springer, Berlin, Heidelberg.
4. Venkatasubramanian, V. (2019). The promise of artificial intelligence in chemical engineering: Is it here, finally. *AIChE Journal, 65*(2), 466–478.
5. Nowakowski, P., Szwarc, K., & Boryczka, U. (2018). Vehicle route planning in e-waste mobile collection on demand supported by artificial intelligence algorithms. *Transportation Research Part D: Transport and Environment, 63,* 1–22.
6. Ahmad, F., Abbasi, A., Li, J., Dobolyi, D. G., Netemeyer, R. G., Clifford, G. D., et al. (2020). A Deep learning architecture for psychometric natural language processing. *ACM Transactions on Information Systems (TOIS), 38*(1), 1–29.
7. Moen, E., Bannon, D., Kudo, T., Graf, W., Covert, M., & Van Valen, D. (2019). Deep learning for cellular image analysis. *Nature methods*, 1–14.
8. Ardabili, S., Mosavi, A., Dehghani, M., & Várkonyi-Kóczy, A. R. (2019, September). Deep learning and machine learning in hydrological processes climate change and earth systems a systematic review. In *International Conference on Global Research and Education* (pp. 52–62). Springer, Cham.
9. Bauer, A., Bostrom, A. G., Ball, J., Applegate, C., Cheng, T., Laycock, S., et al. (2019). Combining computer vision and deep learning to enable ultra-scale aerial phenotyping and precision agriculture: A case study of lettuce production. *Horticulture research, 6*(1), 1–12.
10. Sajeev, S., Maeder, A., Champion, S., Beleigoli, A., Ton, C., Kong, X., & Shu, M. (2019). Deep Learning to improve heart disease risk prediction. In *Machine Learning and Medical Engineering for Cardiovascular Health and Intravascular Imaging and Computer Assisted Stenting* (pp. 96–103). Springer, Cham.
11. Schneider, P., Walters, W. P., Plowright, A. T., Sieroka, N., Listgarten, J., Goodnow, R. A., Fisher, J., Jansen, J.M., Duca, J.S., Rush, T.S. & Zentgraf, M. (2019). Rethinking drug design in the artificial intelligence era. *Nature Reviews Drug Discovery*, 1–12.
12. Justesen, N., Bontrager, P., Togelius, J., & Risi, S. (2019). Deep learning for video game playing. *IEEE Transactions on Games.*
13. Naik, N., & Mohan, B. R. (2019, May). Stock Price Movements Classification Using Machine and Deep Learning Techniques-The Case Study of Indian Stock Market. In *International Conference on Engineering Applications of Neural Networks* (pp. 445–452). Springer, Cham.
14. Lu, C. Y., Suhartanto, D., Gunawan, A. I., & Chen, B. T. (2020). Customer satisfaction toward online purchasing services: Evidence from small & medium restaurants. *International Journal of Applied Business Research, 2*(01), 1–14.

15. Piccione, P. M. (2019). Realistic interplays between data science and chemical engineering in the first quarter of the 21st century: Facts and a vision. *Chemical Engineering Research and Design, 147,* 668–675.

16. Beck, D. A., Carothers, J. M., Subramanian, V. R., & Pfaendtner, J. (2016). Data science: Accelerating innovation and discovery in chemical engineering. *AIChE Journal, 62*(5), 1402–1416.

17. Qi, C., Fourie, A., Chen, Q., Tang, X., Zhang, Q., & Gao, R. (2018). Data-driven modelling of the flocculation process on mineral processing tailings treatment. *Journal of Cleaner Production, 196,* 505–516.

18. Han, H., Zhu, S., Qiao, J., & Guo, M. (2018). Data-driven intelligent monitoring system for key variables in wastewater treatment process. *Chinese Journal of Chemical Engineering, 26*(10), 2093–2101.

19. Almeshaiei, E., Al-Habaibeh, A., & Shakmak, B. (2020). Rapid evaluation of micro-scale photovoltaic solar energy systems using empirical methods combined with deep learning neural networks to support systems' manufacturers. *Journal of Cleaner Production, 244,*.

20. Wu, H., & Zhao, J. (2018). Deep convolutional neural network model based chemical process fault diagnosis. *Computers & Chemical Engineering, 115,* 185–197.

21. Kitchin, J. R. (2018). Machine learning in catalysis. *Nature Catalysis, 1*(4), 230–232.

22. Spellings, M., & Glotzer, S. C. (2018). Machine learning for crystal identification and discovery. *AIChE Journal, 64*(6), 2198–2206.

23. del Rio-Chanona, E. A., Wagner, J. L., Ali, H., Fiorelli, F., Zhang, D., & Hellgardt, K. (2019). Deep learning-based surrogate modeling and optimization for microalgal biofuel production and photobioreactor design. *AIChE Journal, 65*(3), 915–923.

24. Pantula, P. D., & Mitra, K. (2020). Towards efficient robust optimization using data based optimal segmentation of uncertain space. *Reliability Engineering & System Safety, 197,* 106821.

25. Haghighatlari, M., & Hachmann, J. (2019). Advances of machine learning in molecular modeling and simulation. *Current Opinion in Chemical Engineering, 23,* 51–57.

26. Alizadeh, R., Allen, J. K., & Mistree, F. (2020). Managing computational complexity using surrogate models: A critical review. *Research in Engineering Design, 31*(3), 275–298.

27. Miriyala, S. S., Subramanian, V. R., & Mitra, K. (2018). TRANSFORM-ANN for online optimization of complex industrial processes: Casting process as case study. *European Journal of Operational Research, 264*(1), 294–309.

28. Dua, V. (2010). A mixed-integer programming approach for optimal configuration of artificial neural networks. *Chemical Engineering Research and Design, 88,* 55–60.

29. Carvalho, A. R., Ramos, F. M., & Chaves, A. A. (2011). Metaheuristics for the feedforward artificial neural network (ANN) architecture optimization problem. *Neural Computing and Applications, 20*(8), 1273–1284.

30. Boithias, F., Mankibi, M., & Michel, P. (2012). Genetic algorithms based optimization of artificial neural network architecture for buildings' indoor discomfort and energy consumption prediction. *Building Simulation, 5*(2), 95–106.

31. Eason, J., & Cremaschi, S. (2014). Adaptive sequential sampling for surrogate model generation with artificial neural networks. *Computers & Chemical Engineering, 68,* 220–232.

32. Jones, D. R. (2001). A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization, 21,* 345–383.

33. Crombecq, K. (2011.) Surrogate modeling of computer experiments with sequential experimental design.

34. Davis, E., & Ierapetritou, M. (2010). A centroid-based sampling strategy for kriging global modeling and optimization. *AIChE, 56,* 220–240.

35. Gorissen, D., Couckuyt, I., Demeester, P., Dhaene, T., & Crombecq, T. (2010). A surrogate modeling and adaptive sampling toolbox for computer based design. *The Journal of Machine Learning Research, 11,* 2055–8722.

36. Müller, J., & Shoemaker, C. A. (2014). Influence of ensemble surrogate models and sampling strategy on the solution quality of algorithms for computationally expensive black-box global optimization problems. *The Journal of Global, 60*(2), 123–144.

37. Chugh, T., Sindhya, K., Hakanen, J., & Miettinen, K. (2019). A survey on handling computationally expensive multiobjective optimization problems with evolutionary algorithms. *Soft Computing, 23*(9), 3137–3166.
38. Miriyala, S. S., & Mitra, K. (2020). Multi-objective optimization of iron ore induration process using optimal neural networks. *Materials and Manufacturing Processes, 35*(5), 537–544.
39. Zoph, B., & Le, Q. V. (2016). Neural architecture search with reinforcement learning. arXiv preprint arXiv:1611.01578.
40. Elsken, T., Metzen, J. H., & Hutter, F. (2018). Neural architecture search: A survey. arXiv preprint arXiv:1808.05377.
41. Hagan Martin, T., Demuth Howard, B., & Beale Mark, H. (2002). Neural network design. *University of Colorado at Boulder*.
42. Graves, A. (2012). Supervised sequence labelling. In *Supervised sequence labelling with recurrent neural networks* (pp. 5–13). Springer, Berlin, Heidelberg.
43. Bengio, Y., Goodfellow, I., & Courville, A. (2017). *Deep learning* (Vol. 1). Massachusetts, USA: MIT press.
44. Deb, K. (2001). *Multi-objective Optimization using Evolutionary Algorithms*. Chichester, UK: Wiley.
45. Arlot, S., & Celisse, A. (2010). A survey of cross-validation procedures for model selection. *Statistics Surveys, 4,* 40–79.
46. Akaike H, Information theory and an extension of the maximum likelihood principle. In: B. N. Petrov, F. Csáki (Eds.), *Proceedings 2nd International Symposium on Inf.* Theory, Tsahkadsor, Armenia, USSR, September 2:8 (1971) 267–281..
47. FB, P. (1978). *Physical metallurgy and the design of steels*. London: Applied Science Publishers Ltd.
48. Yada, H., Ruddle, G. E., & Crawley, A. F. (1987). Proc. Int. Symp. On Accelerated Cooling of Rolled Steel.
49. Mohanty, I., Chintha, A. R., & Kundu, S. (2018). Design optimization of microalloyed steels using thermodynamics principles and neural-network-based modeling. *Metallurgical and Materials Transactions A, 49*(6), 2405–2418.
50. Mohanty, I., Sarkar, S., Jha, B., Das, S., & Kumar, R. (2014). Online mechanical property prediction system for hot rolled IF steel. *Ironmaking and Steelmaking, 41*(8), 618–627.