Deep neural networks in chemical engineering classrooms to accurately model adsorption equilibrium data

Shubangi Kakkar\textsuperscript{a,b}, Witold Kwapisz\textsuperscript{a}, Christopher A. Howard\textsuperscript{c}, K. Vasanth Kumar\textsuperscript{a,b,}\textsuperscript{*}

\textsuperscript{a} Department of Chemical Sciences, Bernal Research Institute, University of Limerick, Ireland
\textsuperscript{b} Synthesis and Solid State Pharmaceutical Research Centre, University of Limerick, Ireland
\textsuperscript{c} Department of Physics & Astronomy, University College London, London, WC1E 6BT, United Kingdom

**A R T I C L E   I N F O**

Article history:
Received 5 February 2021
Received in revised form 23 April 2021
Accepted 24 April 2021
Available online 27 April 2021

Keywords:
Machine learning
Deep neural networks
Final chemical engineering
Adsorption
Equilibrium data
Regression analysis
Bioprocess engineering

**A B S T R A C T**

The latest industrial revolution, Industry 4.0, is progressing exponentially and targets to integrate artificial intelligence and machine learning algorithms with existing technology to digitalise chemical processes across the industry, especially in the area of online monitoring, predictive analysis and modelling. Machine learning algorithms are being constantly implemented in both academic laboratories and industry to uncover the underlying correlations that exist in the high-dimensional and complex experimental and synthetic data that describes a chemical process. Indeed soon, proficiency in artificial intelligence methodology will become a required skill of a chemical engineer. It is therefore becoming essential to train chemical engineers with these methods to help them to adapt to this new era of digitised industries. Keeping these issues in mind, we introduced deep neural networks to the final-year chemical engineering students through a computer laboratory exercise. The exercise was delivered in fast-track mode: the students were asked to develop deep neural networks to model and predict the equilibrium adsorption of uptake of three different acids by activated carbon at four different temperatures. In this manuscript, we discuss in detail this laboratory exercise from delivery and design to the results obtained and the students’ feedback. In the classroom, the students compared the adsorption equilibrium data obtained using the established theoretical adsorption isotherms and empirical correlations with the neural networks developed in the classroom. The experience obtained from the classroom confirmed this exercise gave the students the essential knowledge on the AI and awareness on the jargons in the world of machine language and obtained the required level of coding skills to develop a simple neural net with one layer or a sophisticated deep networks to model an important unit operation in chemical engineering and to accurately predict the experimental outcomes.

© 2021 The Author(s). Published by Elsevier B.V. on behalf of Institution of Chemical Engineers. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

1. Introduction

The concept of artificial intelligence (AI) dates back to 1965 at Dartmouth College in Hanover, New Hampshire (Lungarella et al., 2007). Although AI can sound like a buzz word for non-specialists, everyone will agree that it significantly influences our daily lives. For instance, AI helps us to automatically sort emails, makes suggestions in Amazon, Netflix/YouTube, Alexa, Siri, and enables chatbots and virtual assistants for banking. An AI system exists in smart TVs mobile apps, and other commercial technologies and its ubiquity are ever-increasing. Today AI-based methods have been applied in many fields including linguistics (Spiro et al., 2017), cognitive sciences (Collins and Bobrow, 1977), medicine (Topol, 2019), neuroscience (Hassabis et al., 2017), engineering (Kalogirou, 2003; Uraikul et al., 2007) and technology (Zang et al., 2015), and market analysis (Trippi and Turban, 1992; Wang et al., 2018).

AI-based methods are now widely employed in the field of chemical engineering by both academicians and industrialists (Himmelblau, 2000; Venkatasubramanian, 2019). For example, AI is widely used as a tool for predictive analysis and has been successfully used to model processes including crystallization (Damour et al., 2010; Velasco-Mejia et al., 2016; Yang and Wei, 2006), adsorption (Kharitonova et al., 2019), distillation (Singh et al., 2007), gasification (Pandey et al., 2016), dry reforming (Azzam et al., 2018) and filtration (Bagheri et al., 2019). Additionally, AI has been used
to predict the properties of fluids (Joss and Müller, 2019), the melting points of crystals (Gamidi and Rasmuson, 2020, 2017), nucleation probability (Hjorth et al., 2019) and interfacial tension (Kumar, 2009). In the field of analytical chemistry, AI has been used for predicting crystal stability (Ye et al., 2018) and X-ray absorption spectra (Rankine et al., 2020), molecular recognition (from the library of ToF-SIMS spectra (Tuccitto et al., 2019)), crystal structure prediction (Ryan et al., 2018) and elemental composition (Ismail et al., 2019). Thus important applications of AI include material design (Mosavi and Rabczuk, 2017), screening of materials for targeted applications (Zhou et al., 2019), drug discovery (Fleming, 2018), drug formulations (Yang et al., 2019), pharmacokinetic modelling (Deshpande et al., 2018) and advanced biochemical analyses such as cancer detection and treatment (Bi et al., 2019; Patel et al., 2020; Wedsor et al., 2020). In chemical engineering, the power of AI is already creating significant impact since it can be easily implemented into existing systems to model and monitor complex processes in real-time and enabling real-time diagnosis and control. In chemical engineering industries and many academic laboratories, AI is already in use, mostly to monitor, predict and control the outcomes of the unit operations. In chemical engineering, AI algorithms are often used in fault diagnosis (Venkatasubramanian, 2011; Venkatasubramanian et al., 2003), process control (Hoskins and Himmelblau, 1992), model chemical reaction kinetics (Molga et al., 2000), monitor unwanted events like crystal agglomeration (Heisel et al., 2019) that can occur during the crystallisation process. The applications of AI in chemical engineering can be found in the review works of Himmelblau (Himmelblau, 2000) and Venkat Venkatasubramanian (Venkatasubramanian, 2019). In the emerging field of material informatics, AI can be exploited to predict material properties from the structure or the structure from the material properties (AIChE ChEnected, 2019; Venkatasubramanian, 2019). For example, neural networks can be used to build new materials with the desired level of properties based on the force fields and structure–property relationships obtained from first principle calculations. Advances in computational power, availability of a wide range of machine learning tools, advances in instrumentation and data acquisition capabilities, together with the access to large datasets in the literature are starting to make AI-based methods more affordable, faster, more accurate and will eventually make AI a mainstream chemical engineering tool. In industry, AI is already being adopted in the fields of drug discovery (Fleming, 2018; Paul et al., 2020) and fault diagnosis of machine failure and prevention (Venkatasubramanian, 2011).

Despite the building momentum, adopting AI methods is still hindered by a lack of knowledge of its implementation and use. In fact, it is arguably now essential to educate engineering students in AI, from understanding its widespread applications and potential for solving engineering problems in real-time to basic methodology and implementation. In long term, chemical engineers with knowledge in AI methods will further expedite the methods from the existing lab-scale projects towards mainstream industrial applications.

Ultimately, AI is the training of machines via imitating the cognitive behaviour of humans. In the classroom, the basics of AI has been introduced to the students by introducing an AI function called deep learning. Deep learning is a branch of AI and can perform actions analogous to a human brain, specifically, it can process given information, analyse, recognize patterns, remember events and can make decisions analogous to a human. Deep learning is more often called as deep neural learning or deep neural network and they are nothing but artificial neural networks (ANN) composed by mathematical neuron or also called as the perceptrons. The perceptron is the main building blocks of artificial neural networks. The artificial neural network (ANN) is a mathematical toolbox containing one or more layers, and each of these layers contains either one or more perceptron. ANN containing more than one layer is called deep neural networks or deep networks. Each perceptron contains a mathematical function and like biological neurons, the perceptron in the ANN, when given some inputs, can communicate with each other, learn, recognize the patterns, correlate the given inputs with the expected outcomes, and at some stage will even make the ANN to think and perform actions like humans or give expected output. Every student in the classroom will have used most common voice assistants such as Google Assistant, Alexa and Siri and they already have a general idea about AI and maybe even aware of artificial neural networks. However, interactions with the students in the classroom show that many have been intimidated perhaps due to media hype and view ANN as a complex, sophisticated, robotic plus intelligent toolbox. Similarly, many students presume that ANNs may be too complex to implement in chemical engineering, especially to write code themselves that allows to construct an ANN.

To date, AI is most widely used in engineering to find the relationship between a dependent variable and independent variables. The algorithms that allow predictions of outcomes of the relationship between a dependent variable and independent variable are often called regression algorithms. Alternatively, AI can be purpose built using classification or clustering algorithms depending on the inputs available about the problem that allow the prediction of the desired output. Such algorithms can be used for pattern recognition i.e. to search and identify regularities in the data. Such pattern recognition (Bishop, 1995; Dougherty, 2012; Rogers and Kabrisky, 1991) is commonly used in diagnosing dangerous diseases (Bezdek et al., 1993; Nithya and Ilanga, 2017), Google Assistant, Amazon’s Alexa, Microsoft’s Cortana and Siri use such algorithms for speech recognition (Kepuska and Bohouta, 2018) by processing a ‘combination of words’ asked to them by the user.

In this manuscript, we describe an exercise that was delivered to the final year chemical engineering students, which aimed to build an ANN using MATLAB that can model and predict the adsorption equilibrium data of three different acids from a fermentation broth using activated carbons at different temperatures. Adsorption depends on a range of experimental conditions including temperature, gas/solute concentration, adsorbent mass and properties such as surface area, pore-volume, pore size distribution and also the properties of the solute itself, like their molecular volume, molecular structure, molecular surface area. It is therefore, an extremely difficult task to extract empirical correlations that allow the prediction of the amount adsorbed as a function of all of these variables. In the classroom, we show the students how AI methodology can identify the underlying relationships between these variables, without requiring the knowledge about the actual physics behind the processes. Although ANN was used here as a predictive/modelling tool to treat the adsorption equilibrium data, the main learning outcome of this exercise is to introduce students the working principles of ANN and to show them the skeletal structure of ANN with mathematical details. Another key objective was to give coding experience to the students and increase their confidence in this area and to view ANNs as an accessible toolbox that can be easily built, effectively trained and tested to solve problems in chemical engineering. This computer-based laboratory exercise can be readily performed within 2–3 h and we believe this exercise is compatible with various aspects in the curriculum of undergraduate/postgraduate chemical engineering and chemistry courses. The applicability of AI can be easily incorporated with several chemistry/chemical engineering courses such as chemical engineering design methods, mathematics for chemical engineers, design projects, process control and the students can even be encouraged to use AI in the final year research projects.
2. History of AI

Humans have always been fascinated with the idea of constructing intelligent machines, robots that can think like humans and make intelligent decisions and exhibit sentient behaviour of humans. Historically, this idea remained a science fiction until 1950, when Alan Turing, the father of modern computer science, explored the idea of using mathematics for AI (Turing, 1950, 1936). In his paper, Computing Machinery and Intelligence, Turing suggested the concept of making the machines think like humans based on available information and then make logical decisions or solve problems (Turing, 1950). He also discussed how to build intelligent machines and to test their intelligence. Five years later, the term Artificial Intelligence was coined by John McCarthy et. al. and for the first time, an AI-based program called the Logic Theorist was designed to mimic the problem-solving skills of a human (McCarthy et al., 2006).

In the classroom, we introduced the history of AI to the students to stimulate their attention on this topic, open their curiosity and show how the field of AI evolved in parallel with the increase of computational power. The continuous and rapid increases in the computational power while the computational cost becomes more and more affordable and accessible, allows the students to imagine the scope and the future of AI. The historical context was introduced to the students without detailing the crises faced by the AI researchers due to lack of support and funding, lack of infrastructure that hindered the growth of AI for a significant period of time. In the class, students were made aware of the works of Alan Turing and shown a graph (see reference (Anyoha, 2017)) depicting the evolution of AI. Snapshots of the first page of the Turing’s articles published in the Proceedings of the Mathematical Society (Turing, 1936) and Mind (Turing, 1950), in the years 1936 and 1950 respectively, were shown to the students. Students were also supplied with the review articles that discuss the history of AI (see references (Lungarella et al., 2007) and (Anyoha, 2017)). During this introduction, it was clear that students were amazed by the history of AI and even become curious about the way AI works and asked several questions on how mathematics can help to build an intelligent machine, which accomplished one of the teaching outcomes.

3. Adsorption equilibrium data: bringing laboratory research to the classrooms

Studying adsorption was selected for this exercise since adsorption equilibrium data are readily available and in plenty. Adsorption equilibrium data of a wide range of adsorbates and adsorbents can be obtained from several published works in established journals like the Journal of Chemical Engineering Data (Da Silva and Miranda, 2013) and from handbooks like the Adsorption Equilibrium Data Handbook (Valenzuela and Myers, 1989). The predictive capability of the neural networks correlates with the accuracy of the training process (this is discussed in detail in the later sections) and the accuracy of the training process can be improved by the amount of data used to train the network. Since experimental adsorption data are available in abundance they can be implemented easily in the classroom environment and students can be easily separated in groups to model the adsorption equilibrium data of a wide range of adsorbents and their adsorption capacity for different target molecules. In our study, students were given the data of adsorption equilibrium data of three different organic acids from their fermentation broth. The overall aim is to build an artificial neural network that can predict the experimental outcomes, which is the amount of (three different) acids adsorbed by the activated carbon at different temperatures. This system was also selected as the exercise was delivered to the final year chemical engineering students registered for the module Bioprocess Engineering. This exercise was also designed to teach students to understand the fundamental principles of adsorption, which is considered to be one of the main downstream unit operation and commonly used to purify the products obtained from biological processes. From the Bioprocess Engineering viewpoint, currently, most of the acids are produced using biological methods and adsorption is considered to be the ideal unit operation for the recovery of acids from the fermentation broth.

The experimental equilibrium data was extracted from a research article published by Silva and Miranda (Da Silva and Miranda, 2013), in the Journal of Chemical Engineering Data. At the end of the exercise, the students were suggested to read the original research article from which the experimental equilibrium data were obtained and three more research articles that are published in the field of adsorption and neural networks (Kumar et al., 2010, 2008c; Kumar and Porkodi, 2009). This background reading not only gives the students an opportunity to read a full-length research article but also aims to give them an idea about how the adsorption techniques are used in research laboratories to provide solutions to industrial problems. In their final reports, the discussion of the contextual literature including the scientific adsorption studies was notably of very high quality, indeed exceeding the average quality of similar exercises. This showed that the students were inspired by the subject matter of both AI methods and adsorption studies.

4. Modelling of adsorption equilibrium data using theoretical adsorption isotherms: regression analysis

Adsorption equilibrium data can be modelled using theoretical expressions like the Freundlich (Freundlich, 1906) and Langmuir isotherms (Langmuir, 1918). For this particular exercise, the students were asked to use a non-linear regression analysis to extract the isotherm parameters. The students were already familiar with linear regression techniques, which they used in other modules like reaction engineering and bioprocess engineering, where they use linear expressions to predict the kinetic parameters involved in the first-order kinetics, second-order kinetic, Michaelis Menten parameters and Monod kinetic constants (Shuler and Kargi, 1992). Most of the students were not familiar with the non-linear regression analysis and thus as part of this exercise, a trial and error method was introduced to the students that allow the isotherm parameters to be obtained. The trial and error method is a straight forward plus a simple technique and can be easily performed using a simple spreadsheet, such as Microsoft Excel.

The trial and error method involves a mathematical iteration procedure, where an error function is optimized to minimize the error distribution between the experimental equilibrium data and the predicted theoretical adsorption isotherm. The objective function is optimized using the solver add-in available within Microsoft Excel. The error distribution can be minimized by adjusting a suitable error function. In this study, an iterative procedure was implemented to minimize the sum of the errors squared (ERRSQ) to minimize the error distribution between experimental data and predicted isotherm. The ERRSQ is mathematically defined as:

\[ \text{ERRSQ} = \sum_n \left( q_{\text{experimental}} - q_{\text{theoretical}} \right)^2 \]  (1)

where, \( n \) is the number of data points in the experimental adsorption isotherm, \( q \) refers to the amount of acid adsorbed at equilibrium, \( q_{\text{experimental}} \) is the experimentally obtained \( q \) value and \( q_{\text{theoretical}} \) is the amount adsorbed at equilibrium predicted by the theoretical adsorption isotherm (either Freundlich (Freundlich, 1906) or Langmuir (Langmuir, 1918)). As an alternate to ERRSQ, other error functions, like the coefficient of determination, \( r^2 \) an...
average relative error can be used to minimize the error distribution (please see the works of Kumar et al. (Kumar et al., 2008a, 2008b)). To perform non-linear regression analysis, the objective function, ERSQ was set to zero and the widely accepted generalised reduced gradient method, available in Microsoft Excel, is used to solve for the isotherm parameters. Non-linear regression analysis relies on an iterative procedure, which requires the initial values for the adsorption isotherm parameters, which are not known a priori. Thus, the initial guess values for the Langmuir (Langmuir, 1918) and Freundlich (Freundlich, 1906) isotherm parameters were obtained by linear regression using the least-squares method. For linear regression, the widely-accepted linearized form of Freundlich (Freundlich, 1906) and Langmuir (Langmuir, 1918) isotherms were used to obtain the isotherm parameters. The original Langmuir and Freundlich expressions and their linearized expression and the way to obtain the isotherm parameters from their slope and intercept are given in Table 1.

![Graph](image)

In Fig. 1, we plot the experimental and the predicted adsorption isotherms of three different acids at 20 °C and we also show the calculated isotherm parameters from Table 1 and the corresponding ERSQ values. In this figure, we show only the theoretical isotherm predicted using the non-linear regression analysis. In the classroom, the students were advised to include the theoretical isotherms obtained using both linear and non-linear regression analysis. In this manuscript, we only show the adsorption isotherms obtained at 20 °C for demonstration, however in their final report, the students were asked to include the adsorption isotherms obtained at different temperatures (20 °C, 30 °C, 40 °C and 50 °C) and the predicted isotherm constants using both linear and non-linear regression analysis. Based on ERSQ values, the Langmuir isotherm closely represents the experimental equilibrium data. For this particular exercise, the students were asked to fit the experimental equilibrium data in the two different two-parameter isotherms. Obviously, it is possible to introduce other theoretical adsorption isotherms with more than two parameters, however, due to the time limitations (the lecture plus tutorial was delivered within three hours), only the two established two-parameter isotherms were used. The obtained isotherm parameters reveal information about the physics of the adsorption process. For instance, the best fit of experimental equilibrium data suggests that the adsorption is due to the monolayer coverage of solute molecules on to the adsorbent surface. According to Langmuir isotherm, there exists a maximum limit which will be equal to the number of molecules adsorbed on to the entire surface of adsorbent; this means all the adsorbent surface will be covered by one layer of solute molecules. To support the readers, in the supplementary file we have uploaded the Microsoft Excel spreadsheet, where we explain in detail how to obtain the Langmuir and Freundlich isotherm parameters using non-linear regression analysis.

![Graph](image)

The adsorption at equilibrium depends on the properties of the adsorbents including surface area, pore-volume, adsorbate properties like their size, their molecular volume and area, presence of functional groups, electrostatics, among others, and this makes it almost impossible to define a unique expression from a theoretical point of view that can successfully correlate all these properties with the equilibrium adsorption uptake. Although the theoretical isotherms can closely represent the experimentally obtained adsorption equilibrium data, the determined isotherm parameters are specific to the adsorbent/adsorbate studied and also the experimental condition like temperature. The complexity of the system means it is not possible to develop any theoretical adsorption isotherm that allows the prediction of the amount adsorbed at equilibrium as a function of temperature and different types of adsorbates.

As shown in the next section, traditional approaches of developing empirical expressions to correlate experimental outcome as a function of the operating variables/solute properties (like initial concentration, temperature, solute molecular volume, solute surface area) often suffer poor accuracy due to the highly complex non-linear relationship that between the equilibrium adsorption uptake and the operating variables. It is here that machine learning approaches can provide a solution and can correlate the complex...
and highly non-linear relationship that may exist between the systems’ parameters with the adsorption uptake.

5. Empirical correlation approach using non-linear regression analysis

In chemical engineering, it is common to develop empirical correlations in order to predict an experimental outcome as a function of operating variables/experimental conditions. However, the error distribution between the experimental data and the predicted experimental outcomes are often high. Empirical correlations have been used to predict the solution properties like boiling point (Joss and Müller, 2019), crystal growth kinetics (Vasanth Kumar et al., 2008), interfacial tension (Kumar, 2009), and melting point (Gamidi and Rasmuson, 2017; Habibi-Yangjeh et al., 2008; Karthikeyan et al., 2005; Torrecilla et al., 2008). For the case of adsorption, empirical correlations have been used to calculate the multicomponent adsorption equilibrium data for the combination of three different basic dyes (McKay and Al Duri, 1989). Other well-known correlations include expressions used to predict the mass and heat transfer coefficients. In the classroom, the students were asked to develop an empirical expression as in Eq (2) to correlate different parameters like initial concentration ($C_0$), temperature ($T$), molecular surface area of the adsorbate (MSA), molecular volume of the adsorbate (MVA) with the amount adsorbed at equilibrium conditions:

$$q_e = a[C_0]^b(T)^c(MSA)^d(MVA)^e$$

The empirical constants, $a$, $b$, $c$, $d$ and $e$ can be obtained by the above-described non-linear regression analysis. To maintain consistency, while implementing the trial and error non-linear regression analysis, the students were asked to solve the above expression with the same initial guess values. The initial guess values were obtained from the power trendline that best fit the data in the plot of $q_e$ versus the variables. In the supplementary file, we have uploaded the Microsoft Excel spreadsheet, where we explained in detail how to get the constants in the empirical expressions (as in Eq 2) using non-linear regression analysis. According to the non-linear regression analysis, Fig. 2 shows the parity plot, where the $q_e$ values obtained from Eq (2) $q_{e,\text{experimental}}$ are plotted against the $q_e$ values obtained through experiments. In Fig. 2, we also give the empirical constants in Eq (2) obtained using the non-linear regression analysis. A correlation that accurately predicts the experimentally measured $q_e$ value should yield points on the diagonal line of the parity plot. However, it is evident from Fig. 2 that the developed empirical correlation using a non-linear regression analysis approach poorly predicts the equilibrium adsorption uptake. As shown in the next section, ANN can prove to be effective while solving this type of problems.

The purpose of developing an empirical expression in the classroom is to show how the non-linear regression analysis can be used to develop engineering quality correlations. Another main purpose is to show the students that the empirical correlations can produce unsatisfactory results and often predict the experimental outcomes with larger errors. Additionally, the non-linear regression analysis implemented in the classrooms also require initial guess values and depending on these guess values, the outcome changes and clearly, there is no universal procedure to obtain these initial guess values. Nevertheless, once the parameters were determined, the developed expressions can be used to predict the adsorption uptake.

6. Neural network basics and architecture

A human brain contains several billion neurons (Herculano-Houzel, 2009). A biological neuron contains three main components, dendrites, soma and an axon (see Fig. 3a). The dendrites help receive information or signals from other neurons. Synapses connect the axon of the one neuron to the dendrites of the other neuron. The signals are transmitted via the synaptic gap by means of a chemical process. The synapses determine the weight of the information received from other neurons and modify the incoming signal. The soma or the cell body sums all the received signals or the weighted inputs. When the sum of the weighted inputs exceeds a threshold the cell fires a signal over its axon to other cells (Faussett, 2006).

ANNs are mathematical models that represent the behaviour of neurons found in the human brain. ANNs contain one neuron or more than one neurons connected to each other via a pattern which defines their architecture. The properties of the artificial neuron also called perceptron (see Fig. 3b) are the single processing units that compose the ANN as suggested by the properties of the biological neurons. Similar to biological neurons, the processing elements in a perceptron receive many signals as input. The input signals may be modified by a weighting at the receiving synapse. The processing elements will sum the weighted inputs and pass them into an activation function also called the ‘propagation function’. When sufficient input is received, the neuron transmits a single output, which may go to many other neurons, similar to the axon branches of a biological neuron.

In Fig. 3c, we show the mathematical representation of a neuron or the single processing unit in the ANNs. The neuron receives the signal from an input vector $p$ that contains $n$ elements. In Fig. 3c, $p_1, p_2, p_3, \ldots, p_n$ represent the individual elements or individual inputs. These individual inputs will be multiplied by respective weights, $w_{1,1}, w_{1,2}, w_{1,3}, \ldots, w_{n,1}$. The weighted inputs will then be fed to the summing junction and their sum will be equal to $w_p$. The processing unit or the neuron has a bias $b$, which will be added to the weighted inputs to form the net input $i$ is given by:

$$i = w_{1,1}p_1 + w_{1,2}p_2 + \ldots + w_{n,1}p_n + b = Wp + b$$

The net input $i$ will be sent to the transfer function $f$ to get the neurons output $a$, that can be mathematically written as:

$$a = f(i) = f(Wp+b).$$

In MATLAB, different transfer functions are included in the Neural Network Toolbox (Demuth et al., 1992). The most commonly used transfer functions are hardlim, purelin, tansig and logsig and...
Fig. 3. (a) Model of a biological neuron sketched using 3D model tool in Microsoft Presentation (b) oversimplified version of a perceptron (c) mathematical representation of a perceptron (d) a feedforward network with n inputs and one hidden layer that contains δ number of neurons (e) a feedforward network with n inputs and two hidden layers that contains δ number of neurons in each layer and (f) shows the supervised training strategy used to train the feed-forward artificial neural network.
the neurons may use any one of the transfer functions to generate the output. In the classroom, the students were encouraged to refer to the Neural Network Toolbox manual of the MATLAB (Beale et al., 2010) for additional details about the transfer functions available in this toolbox (Demuth et al., 1992). It should be mentioned here that, if we have only one neuron, the output \( o \) will be a scalar quantity. If we have more than one neuron, and if the output of the first neuron is connected to a second neuron, then the output from the first neuron is a vector.

Most of the neural networks that are used to solve chemical engineering problems contain more than one layer, and each layer contains more than one neuron (or perceptron). Irrespective of the number of neurons or the number of layers in the neural network, the working principle is essentially the same. In Fig. 3d, we depict a neural network that contains only one layer, but this layer contains \( \delta \) number of neurons and \( n \) number of input elements. Each neuron receives signals from the input vector \( p \) that contains \( n \) elements. Each of the inputs will be multiplied by a weight and the weighted inputs will be fed to the summing junction in each neuron. In each neuron, a bias will be added to the weighted inputs to the net input and will be sent to the transfer function to get an output from each neuron \( (o_1, o_2, o_3, \ldots , o_n) \) as shown in Fig. 3d.

The network as shown in Fig. 3b–d) are called feedforward network or backpropagation networks. A feedforward network might contain either one layer or more than one layer. A typical feedforward ANN will contain inputs, outputs and either one or more than one layers are connected in between the inputs and outputs. In the classroom, the students were asked to develop feedforward networks with multiple layers. In the feedforward network, the different layers are connected in series and the information are fed only in the forward directions and thus named feedforward ANNs.

In Fig. 3e, we show the typical structure of a feed-forward ANN that contains 5 layers. If there is more than one layer in between the inputs and outputs, the outputs from the neurons in the preceding layer will become the input vector to the neurons in the next hidden layer. If there is more than one layer in between the inputs and outputs, the layer which produces the final output is called the output layer. For instance, the output of the ANN in Fig. 3d can be connected to one more hidden layer followed by an output layer as shown in Fig. 3e. Fig. 3e shows how the information flows from one layer to another layer. The outputs from the first hidden layer will become the inputs to the neurons in the second layer. Likewise, the outputs from the second layer will become the inputs for the final output layer that contains only one neuron. The output from the final layer, \( o_f \) will be the net output of the constructed ANN. In Fig. 3e we also show the weights and the bias associated with each neuron in all the layers.

Once the network is built, it is essential to train the network. The training process simply refers to the process of repeatedly feeding the inputs and outputs, followed by adjusting the weights and biases using a suitable algorithm, until the network approximates the propagation function and successfully predicts the outputs for the given set of inputs. Briefly, while training the ANN, both input elements and the corresponding output will be fed into the network (see Fig. 3f). The network will adjust the weights and bias and produce an output. This ANN-obtained output will be compared with the actual output values produced by the propagation function. Training the networks using both inputs and the corresponding target values obtained from the experiments is called supervised learning. If the value of the ANN-predicted output and actual output is high, then the networks weights and bias will be adjusted. This process will be repeated by repeatedly feeding the inputs and the outputs until the network predicts the actual output with high-level accuracy. The process of repeatedly feeding the inputs and outputs to the network is called iterations or epochs in the MATLAB toolbox. The accuracy of the network training also depends on the number of the input/target pairs used to train a network. The larger the data used for training, the greater the accuracy of the network. In our study, the experimental equilibrium data was obtained from the works of Silva and Miranda et al. (Da Silva and Miranda, 2013). For this study, training of the feedforward network was performed using the Levenberg–Marquardt training strategy. The mathematics and logical details on the incorporation of Marquardt’s algorithm into the back-propagation algorithm is explained elsewhere (MacKay, 1992). The learning process can be stopped by the user or it is automatically stopped once the mean squared error between the experimental outcome and the ANN-predicted values reaches a threshold value (the default value in MATLAB is \( 10^{-3} \)). Training of the neural networks by the Levenberg–Marquardt algorithm (Hagan and Menhaj, 1994) is sensitive to both the number of layers, the number of neurons in each hidden layer and the propagation or active function used in each layer. The rule of thumb in ANN is that the higher the number of neurons, the better the predictive power of the network. A successfully trained network, not only should accurately predict the outputs for the given set of inputs used in the training process but also should predict the output for the new inputs that were kept unaware of the network during the training process. This can be tested by asking the network to predict the outputs for new inputs (referred to as a ‘testing set’). In the classroom, the experimental equilibrium data of three different acids on to activated carbon at different temperatures was supplied to the students. The students were asked to manually segregate the data into training and testing datasets. Roughly 10–20 % of the data was used for testing and the remaining percentage of the dataset was used to train the networks. The students were asked to develop a network with hyperbolic tangent sigmoid function in the hidden layer and linear function in the output layer. Furthermore, both the input vectors and the output vector were normalized before the training process, such that they fall in the interval of 0–1, so that their standard deviation and mean will be below the value of 1. In the classroom, the students performed the data segregation, data normalisation using Microsoft Excel. A model spreadsheet with experimental data, normalised data, validation dataset and the training dataset used in the classroom is provided in the supplementary file in which we also explain how to perform the data normalisation.

The completely trained network does not always accurately predict the correct output for the given set of input values, which are kept unaware of the ANN during the training process (ANN size refers to the size of the ANN measured in terms of the number of neurons and the layers that compose the entire network). For instance, a completely trained network can poorly predict the output, when the ANN was supplied with new inputs that were kept unaware of the network during the training process. The accuracy of the ANN depends on the ANN size, type of activation function used and the training period itself. Overfitting refers to exceeding some optimal ANN size, which may finally reduce the performance of ANN in predicting the target values (Tetko et al., 1995). In other words, the network contains too many neurons or parameters than the required number in order to predict the target value in both testing and training datasets. Overfitting can be identified from a large error between the experimental and the ANN-predicted adsorption equilibrium data for the new input data. Overfitting can be eliminated using a trial and error procedure while building the network. To do this, it is essential to separate the dataset into training and testing data set. Once the network is trained using the training dataset, the trained network should be simultaneously tested for its accuracy to predict the outputs for the new inputs that were kept unaware during the training process. The training process should be started with a minimum of one hidden layer and one neuron in that hidden layer, followed by testing of the network. The optimal architecture, which successfully predicts the output for the
The neural network toolbox Version 7 of MATLAB (Beale et al., 2010) (Mathworks, Inc.) was used for simulation. The code used in the classroom to construct the neural network is given in in Box 1. The code given in Box 1 is self-explanatory and straightforward to implement and contains only a few lines of code that can be taught to the class within 1 h. The code given in Box 1 requires the students to perform some basic tasks like data normalisation, data segregation manually. In the classroom, students were asked to modify the code in Box 1 to change the number of hidden layers, the number of neurons in hidden layers and the activation function in each layer. Initially, the students were asked to train the network using the Levenberg-Marquardt algorithm (Hagan and Menhaj, 1994) using the trainlm function available within the MATLAB. In Fig. 4, we show the parity plot, of the ANN-predicted $q_e$ values against the $q_e$ values obtained via experiments For this work, we constructed two different neural networks, the first one containing only one hidden layer and the second one that can be called a deep neural network, containing two hidden layers. The first neural network contains 10 neurons in the hidden layer and one neuron in the output layer. The second neural network contains 6 neurons in the first hidden layer, 3 neurons in the second hidden layer and one neuron in the output layer. A pure linear function was used in the output layer and hyperbolic tangent sigmoid function was used for the neurons in the hidden layers. The students were asked to label the network based on their architecture. For instance, the first and the second network should be labelled as 4–10–1 and 4–6–3–1. Where the 4–10–1 refers to the number of inputs - number of neurons in hidden layer -number of neurons in output. In Fig. 4a and b we show the ANN predicted $q_e$ values for the given inputs in the training and testing datasets, respectively. The predicted $q_e$ values from a properly trained ANN should lie on the diagonal line of the parity plot. It is clear from Fig. 4a that both of the ANNs accurately predict the equilibrium adsorption uptake for the given inputs in the training dataset. To analyse the accuracy of the ANNs, we calculated the

Fig. 4. Parity plot between the experimentally obtained equilibrium adsorption uptake and the equilibrium adsorption uptake predicted by the neural networks trained using the Levenberg-Marquardt algorithm (a) for the given inputs in the training dataset and (b) for the given inputs in the testing dataset. Parity plot between the experimentally obtained equilibrium adsorption uptake and the equilibrium adsorption uptake predicted by the neural networks trained using the Bayesian regularisation algorithm (c) for the given inputs in the training dataset and (d) for the given inputs in the testing dataset.

given set of inputs in training and testing dataset, can be taken as the optimal ANN size.

In the classroom, while training the network, the students were encouraged to change the number of neurons in the hidden layer and even the number of hidden layers while optimizing the transfer function for the given input and output vectors in order to avoid overfitting. For the given problem, the students were advised to use a hyperbolic tangent sigmoid function in the hidden layer and a linear function in the output layer. In many cases, such a network can be trained to approximate any function and the predictive power can be obtained with close to 100% accuracy. In fact, this architecture is more than enough to predict the adsorption equilibrium data with high-level accuracy. Nevertheless, in the process of building the network, to test the power, flexibility and simplicity of ANN, students performed several trials by manually increasing the number of layers and the neurons in the hidden layers to find a network that successfully predicts the targets for the inputs in the training and testing dataset.
coefficient of determination values, $r^2$ values between the values and the values obtained from experiments and the ANN predicted outcomes. If the model is 100% accurate, then $r^2$ is equal to 1. The $r^2$ values are automatically generated by MATLAB, or alternatively, it can be obtained separately from the formula (please see the Microsoft Excel spreadsheet in the supplementary information, where we showed how to get the $r^2$ values; this sheet was supplied to the students during the class hours) given below:

$$r^2 = \sum \left( \frac{q_e \text{experimental} - q_e \text{calculated}}{q_e \text{calculated}} \right)^2$$

(5)

Where $q_e$, calculated refers to $q_e$ values obtained from the empirical expression, theoretical expressions like Langmuir, Freundlich or by the neural network. For both the networks, the coefficient of determination ($r^2$) between the experimentally-obtained $q_e$ values and the ANN predicted values was > 0.98. This indicates that the ANN is fully trained. It should be mentioned here that the accuracy of the neural network can be improved by adding more input and the number of data points. In this study, we used only four inputs (see Box 1) and 75 data points. The quality of the ANN can be improved by adding new inputs, such as the molecular weight of the adsorbates, adsorption energies, properties of adsorbents like surface area and pore volume, etc. Nevertheless, the $r^2 > 0.98$ is reasonably acceptable, especially if we compare the results with the ones obtained from the empirical correlation (see Fig. 2, the $r^2$ between the experimentally obtained $q_e$ values and the $q_e$ obtained from Eq. (2) was significantly low and <0.75).

A fully trained network must be robust and should predict the experimental outcome for the new inputs. Both of the networks described above predict the amount adsorbed at equilibrium for the new inputs with reasonable accuracy. It can be observed from Fig. 4b that the constructed networks predicted the equilibrium adsorption uptake of all the three different acids with reasonable accuracy. Most of the predicted values fall within the 15% error limit. This may appear slightly disappointing if we consider the potential of neural networks to generalise many complex problems. For instance, in the field of chemical engineering, ANNs were proven to accurately (close to 100% accuracy) predict crystal growth kinetics and adsorption kinetics (Kumar, 2009; Kumar and Porkodi, 2009; Vasanth Kumar et al., 2008). However, it should be remembered that neural network accuracy can be improved by training the network with more inputs and additional data points whenever new data is available. There is always a room to improve the accuracy of the ANN, which can be done by modifying the network structure and the propagation functions used. The current exercise was not delivered to the students with an intention to construct a network that can model and predict the adsorption equilibrium data with very great accuracy, but rather to give the students the knowledge to construct different types of networks, adjust the network structure, feed the network with experimental data, adapt different training strategies and explore the avenues to improve the quality of the network to predict the desired targets. In the classroom, the students constructed several networks and tested their accuracy. The results in Fig. 4 are obtained from a few of the many network architectures they constructed.

As mentioned earlier, training the network using the Levenberg-Marquart strategy is sensitive to the number of neurons and the number of hidden layers and often suffers from overtraining and overfitting. Thus once the students were familiarised with the supplied code (see Box 1), they were shown how to implement a Bayesian regularisation technique in combination with the Levenberg-Marquardt’s algorithm (Hagan and Menhaj, 1994). Bayesian regularisation technique avoids both overtraining and overfitting and the algorithm works best if the network’s input and output are scaled within the range of −1 to +1 (Demuth et al., 1992).

Bayesian regularisation modifies the performance of the transfer function and reduces the overall noise thus mitigating the problem of overtraining and also overfitting (MacKay, 1992). This method also automatically stops the training process once the algorithm is truly converged. Implementing this algorithm is very straightforward in MATLAB (see Box 1). The algorithm can be considered as truly converged when the network can memorize the training

Box 1: MATLAB script used in the classroom to build a deep neural network.

function cg4017BioprocessEngineering2UL%Module name %This code can be used to model adsorption isotherms using feedforward (deep) neural networks %Inputs are initial concentration, temperature, solute molecular surface area, solute molecular volume %Outputs must be normalised so that the inputs will fall within the range 0–1. Please see the Microsoft excel spreadsheet given in supplementary file where we showed how to normalise the data.

input = [copy and paste the input data here from the Microsoft Excel spreadsheet]; %use the training dataset %target = amount adsorbed (we have only one output) %target must be normalised target=[copy and paste the output data here from the Microsoft Excel spreadsheet]; %use the training dataset net = newff(minmax(input), [20 3 1], {'tansig', 'tansig', 'purelin'}, 'trainlm'); % The network will be trained using a Levenberg-Marquardt strategy. Alternatively, ‘trainlm’ can be replaced with ‘trainbr’ to train the network using the Bayesian regularisation algorithm.

net.trainParam.epochs=10,000; %number of epochs or iterations %Training can be stopped by the user once the Mean Squared Error value reaches 10^-6 to 10^-7.

net=trainParam.Ir=0.01; % learning rate

net.trainParam.mc=0.6; %momentum

net=train(net,input,target); %To initiate the training process output = sim(net, input); %This command will generate the output values predicted by the ANN for the inputs in the training dataset.

[output] %Will print the output in the command window of MATLAB.

[target, output]; %Generate a plot of predicted output values versus the target values which are nothing but the normalized $q_e$ values obtained from the experiments

plot(target, ’o’)

hold on

plot(output, ’+’); %A good fit or if the network is completely trained, all the + symbols in red should overlap with blue circles.

%Now we can test the network for its predictive capability input_testingset=[copy and paste the input data here from the Microsoft Excel spreadsheet]; %Use the testing dataset

output_testingset=sim(net,input_testingset); %This command will use the trained network to predict the output values for the new inputs in the testing dataset.

[output_testingset] %To print the normalised outputs (in the command window) predicted by ANN for the %new inputs in the testing set. These values can be copied to the excel sheet and a parity plot can be generated.

view(net); %Generates a good quality figure of the constructed ANN.
examples and simultaneously can generalize the networks so that it can successfully predict the output for the new inputs in the testing dataset. Additionally, the Bayesian regularisation provides a measure on the number of weights and biases effectively used by the network. In contrast to the Levenberg-Marquardt algorithm, which requires guesswork on the ANN size, the Bayesian algorithm (MacKay, 1992) effectively uses and decides the number of network parameters. In fact, while training the network only using the Levenberg-Marquardt algorithm, the students were asked to stop the training manually once the objective function ‘mean squared error’ reached 10−5. User-implemented stopping of training can be performed using the graphical user interface (GUI) available within the MATLAB. In fact, this process itself can alter the accuracy of the network while predicting the outputs for new inputs and sometimes it requires expertise to know when to manually stop the training. These issues can be avoided while implementing the Bayesian regularisation procedure during the training process. For consistency, while implementing the Bayesian algorithm, the students were asked to use the hyperbolic tangent sigmoid function in the hidden layer and linear function in the output layer. The students were recommended to use only one hidden layer and one output layer and asked to gradually increase the number of neurons in the hidden layer starting from one. After several trials, the network with 5 neurons is more than enough to predict the adsorption equilibrium data with reasonable accuracy for the given set of inputs in both training and testing datasets. The students observed that, while implementing the Bayesian regularisation procedure, increasing the number of neurons >5 or increasing the number of hidden layers does not significantly improve the performance of ANN in predicting the adsorption uptake at equilibrium. As the Bayesian algorithm automatically uses the number of weights and biases, irrespective of the size of network parameters, increasing the number of hidden layers >5 should not affect the effective number of weights and biases. This hypothesis was tested by adding one more hidden layer with five neurons. However, the effective number of parameters required to optimize the structure remains unchanged. The number of effective parameters used can be observed by manually increasing the number of neurons in the hidden layer and monitoring the effective number of parameters during the training process. The GUI of the Neural Network Toolbox in MATLAB allows the monitoring of the number of parameters being used while training using the Bayesian regularisation algorithm. Bayesian regularisation procedure confirms that a network with ~16–17 network parameters is sufficient enough to optimize 75 combinations of inputs (training data set) to predict the adsorption equilibrium data of three different acids by activated carbon. For demonstration purposes in this work, we present the results obtained from two different networks trained using the Bayesian regularisation algorithm. The first network labelled as 4-20-1 contains only one hidden layer with 20 neurons in the hidden layer. The second network labelled as 4-20-3-1 contains two hidden layers with 20 neurons in the first hidden layer and 3 neurons in the second hidden layer. These two network architectures were deliberately selected since they contain more than the required number of neurons in hidden layer to optimize the propagation functions. Selecting this architecture will let the students to observe the fact that the effective number of parameters will remain the same while implementing the Bayesian algorithm. Fig. 4c and d show the parity plot of experimentally measured equilibrium adsorption uptake and the values predicted by the 4-20-1 and 4-20-3-1 ANNs (trained using a Bayesian algorithm for the given inputs in training and testing dataset, respectively. However, irrespective of the number of hidden layers or the number of neurons in the hidden layer, the Bayesian algorithm avoids overtraining and only used ~16–17 network parameters to fully train both the networks. In the classroom, the students tested this by constructing a sophisticated deep neural network with five to six hidden layers with 5 neurons in each layer. For the given inputs in the training dataset, the predicted values by both the ANNs are similar as expected since the number of effective parameters remains the same, irrespective of their architecture. In terms of accuracy, most of the data fall on the diagonal line, which means the constructed network is fully trained. The coefficient of determination between the experimentally determined and the ANN predicted qe values was found to be >0.99. The trained networks also successfully predicted the qe values for the new inputs in the testing dataset. In terms of accuracy, for most of the inputs, the percentage error between the experimental data and the ANN predicted values was minimal as most of the values fall on the diagonal line and for some of the inputs the percentage error was slightly higher than 10 % (see the error line in Fig. 4c & d, for guidance we show the 10% error line in these figures). As mentioned earlier, the accuracy can always be improved, by adding more inputs and data points. Nevertheless, it is clear from this exercise that ANN is a very powerful predictive tool and if properly constructed and implemented, it can be used as a modelling tool and can be used to predict the experimental outcomes as in chemical processes. For instance, if we examine Fig. 4c, it is clear that all the points fall on the diagonal line, this means, 4-20-1 ANN model is more than enough to accurately model the adsorption equilibrium data recorded at different temperatures. In this way, instead of using ANN to predict the experimental outcomes for new inputs, the method can be used as a modelling tool and can replace the theoretical adsorption isotherms.

7. Student feedback and conclusions

The main purpose of this laboratory exercise is to make the students engage with the software, understand the logic and basic mathematics of the ANN, and more importantly to make the students aware of the jargon in the field of artificial neural networks and to gain the confidence to use this technique to solve other chemical engineering problems. In addition, the student does not require any coding experience or hands on experience with the Neural Network Toolbox of the Matlab. The codes which are used in this particular exercise are simple and can be taught to the students while delivering this exercise. We provided the basic MATLAB codes to the students in the classroom and encouraged them to modify the codes (e.g., we ask the students to change the propagation function, the number of neurons in the hidden layer, the number of hidden layers, the number of iterations etc.) while building the ANN. The neural network can be quickly built and can be easily taught to the students using the MATLAB Neural Network Toolbox’s graphical user interface and treating them as a purely black-box model. For instance, it is easier to introduce students to ANNs as an (i) architecture composed by different boxes placed under different columns or called layers, (ii) the information flows from one side of the box to another side of the box, (iii) each of these boxes contains a mathematical function, (iv) information exchange occurs between the boxes and the information are weighted or modified at each neuron. (v) ANN can be trained to remember your outputs for the given set of inputs, (vi) once trained ANN can even predict the outputs for any new inputs that are kept unaware of the ANN during the training process and so on. However, this approach can only portray the black-box nature of the ANNs and thus will take away the opportunity to learn about the inner workings of the ANN from the students and thus kill the expected learning outcome.

Another key objective was to introduce the students the basics of ANN and to directly demonstrate its power to solve one chemical engineering problem, which is to predict the equilibrium adsorption of three different acids adsorbed adsorption equilibrium by activated carbon at four different temperatures. The intention was
not to load the students with heavy mathematics involved behind the ANN but to introduce the ANN terminologies, present them the right level of information, the working principles of ANN, the nomenclature of the algorithms and strategies used to train the network. We did this systematically, by first briefly introducing the students to the topic of adsorption and what to the exercise was designed to predict using so-called ‘deep neural networks’ for the first five minutes. We then explained to the students, why as chemical engineers, we learning about AI is important including briefly how chemical engineers solve problems using mathematical expression and the common limitations of these expressions. Then we explained how AI and artificial neural networks can identify the hidden complex and non-linear relationships that can exist between the operating variables and the experimental outcomes. We also talked about the AI, machine learning and deep neural networks followed by the history of the AI, the structure of a biological neuron and finally we introduced the structure of a mathematical perceptor.

We found that it is essential that, during the three hour tutorial period, the teacher/instructor reassure the students repeatedly that, ‘deep neural networks is a straightforward topic and building a neural network is easy to perform. Also that the only prior information required is what type of inputs we need in order to predict an output, the number of hidden layers to start with and the number of neurons in each hidden layer, the propagation function in each layer and the training procedure to be used’. This approach definitely helped to remove the ‘fear factor’ from students who are not comfortable with programming languages. For this exercise, we started the topic of AI from scratch as the students did not have any prior knowledge on the working principles of AI or the mathematical structure of the perceptor. Based on our classroom experience, we found that by the end of the exercise the student has realised that the topic of deep neural networks is not complex and can be executed with a simple MATLAB code that contains less than ten lines. To assist the teachers who may be interested to deploy neural networks in their classrooms, we have uploaded our PowerPoint file, with and without voice over narration, as a supplementary information. In the PowerPoint file, we explain how to deliver this lesson in less than three hours and more importantly how to introduce the topic of deep neural networks in the most simplest possible way. In addition, we are also uploading a separate file, ‘Additional tips to the teachers’, where we explain how the topic of AI and deep neural networks can be delivered to the students. The PowerPoint and the ‘Additional tips to the teachers’ are designed to complement each other. In the PowerPoint, we explain how this exercise was delivered in our classroom and how it can be delivered in fast-track mode.

We recommend that the teachers provide the students with the Neural Network Toolbox Manual, as a standard reference book. The manual contains information about how to use neural networks for pattern recognition, data fitting, and data clustering. If the students are are curious then they can use this document to self-learn to use the neural networks to solve other type of problems, which was not taught during this exercise.

At the end of the exercise, the students were asked to submit a laboratory report after a four-week deadline. This four-week deadline was found to be enough to repeat the exercise delivered during the three hour tutorial period, read the literature and also to gain more theoretical knowledge about the deep neural networks and write the report. To guide the students, we offered support through email and (although, the students completed the laboratory exercise and submitted the report without any further assistance) we also gave two of the best relevant review articles (Himmelblau, 2000; Venkatasubramanian, 2019). In the final report, students were asked to compare the results obtained from the neural network with an empirical relation (see Eq. (2)). To support the students, we provided the Neural Network Toolbox manual. The code which was supplied to the students is more than enough to repeat the exercise at their own pace in their home, build new neural networks with different architecture with different propagation functions in the hidden and output layers. The final reports submitted by the students, clearly showed that the students captured the scientific background and the working principle of the neural networks.

The learning outcome was evaluated based on the student’s final report. In the final report, we asked the students to include an introduction about artificial neural networks, a small chapter about industry 4.0 and digitisation of chemical industry, and the history of neural networks. The literature required to write about the history of artificial neural networks and industry 4.0 was sent to the students via email. We gave clear instructions to the students to build at least five, and up to ten, different neural networks. These can be easily built by simply modifying the number of hidden layers, number of neurons in the hidden layer and the student must include the final architecture of each deep neural network in the final report (the structure of the neural network will be automatically generated by the MATLAB and the student were advised to copy paste these images in the final report). We also recommended that the students train one particular neural network model with two different training algorithm taught in the classroom. The students were also asked to compare the performance of the neural network trained by Levenberg-Marquardt algorithm and the Bayesian algorithm and to include the graph of means squared error versus number of iterations for these. The student must then discuss which training algorithm better predicts the output value for the given inputs in the testing set based on a parity plot. We asked the students to create a table that should include the following parameters, the structure of the network (Say for e.g., 10-10-1 network), propagation function used in the hidden layer, propagation function used in output layer, mean squared error between the experimental data and ann predicted values in the training set, mean squared error between the experimental data and ann predicted values in the training and the testing set. Finally the students were required to identify one particular neural network that better predicts the amount adsorbed at equilibrium. In each of the reports, students discussed the results obtained from at least ten different ANNs that differ by their architecture and each of the network was trained using the algorithms discussed above. Many of the reports discuss the training network using Bayesian algorithm and how it always uses a constant number of weights and biases to optimize a network that contains a specific number of inputs and data points in the training set. Many students modified the propagation functions of the neurons in both the hidden and output layer. The Final report was also checked for any innovations from the students, for example an attempt by a student to add extra inputs such as molecular weight of the adsorbates and their adsorption energies. A few students even divided the given data in to three different sets, one training set and two validation set. This clearly shows that this exercise not only helped the students to build ANNs but also guided them to accurately capture the working principles, mathematics and the logic behind the neural network models and thus achieved the expected learning outcome. We strongly believe that this exercise changed the earlier perception of students about the neural networks as complex black-box models and provided essential knowledge about the inner workings of the neural networks. The guidelines that were given to the students on ‘what should be included’ and ‘how to prepare the final report’ is provided in the supplementary information.

The students were asked to submit their feedback on this exercise together with their final report. In the feedback, we asked the students about the learning experience (open to their own interpretation) and what they thought about the topic of AI in general.
having completed the exercise. We also asked the students to comment on learning about AI as part of their chemical engineering module. We also directly asked the students to feedback on the difficulty level of this topic and to list few chemical engineering problems where a deep neural network could be deployed. Together with the feedback, we asked the students to identify chemical engineering unit operations which can be modelled using the deep neural networks. They also need to identify at least four (or less or more) parameters that can be taken as a representative inputs for the neural network and should provide a rationale for selecting those parameters as inputs. For, e.g., a deep neural network can be used to model gas adsorption equilibrium data. For this case, the students should clearly identify the key parameters that can be as inputs to train the neural network that can predict the target, which is the amount adsorbed at equilibrium. The gas adsorption capacity depends on the adsorbent surface area, adsorbent pore volume, helium density, relative pressure, temperature, molecular weight of the gas, bulk density of the adsorbent and presence of any functional groups on the surface. Selecting these parameters should capture the effect of these parameters on the equilibrium uptake. This will give an idea about the student's level of understanding on the topic delivered plus to make the realise the potential of the deep neural networks to solve different chemical engineering problems.

The feedback from the students about the learning experience was remarkably positive. None of the students in the classroom found this topic difficult. After this exercise, four students used artificial neural networks for their final-year research projects to model three key issues in the field of chemical/environmental engineering. These students used neural nets to: predict the crystal growth kinetics of twenty different pharmaceutical compounds, predict the methane storage capacity of a different class of porous materials, predict the CO₂ selectivity of the adsorbents and quantify the agglomeration degree and crystal breakage during the crystallisation process. Their direct and independent implementation of the methodology in their research projects was an extremely satisfying result it shows the exercise not only taught the neural network modelling as a mathematical technique, but the students realised the power of this tool to solve real-world problems. Additionally, in the research projects listed above, the students modified the code used in this exercise and all the calculations were performed in using desktop computers with minimum configuration. Despite the fact that the whole exercise was delivered in less than three hours, the students still felt that the MATLAB code is simple, easy to understand, easy to modify and does not require any heavy computational power as most of the calculation can be performed with minimum configuration (this might change depending on the strength of the data used in training). We therefore, believe the exercise can be delivered in fast track mode.

This exercise was delivered to the chemical engineering students who are already familiar with MATLAB and have basic coding experience and a strong background in engineering mathematics. However, we propose that this exercise can be delivered to chemistry undergraduate students who might not have any coding experience or never exposed to the MATLAB environment and might not have a strong mathematical background. We recommend the teachers to use the PowerPoint supplied with this manuscript, while introducing neural networks to students who may not have an engineering background. In fact, the neural network toolbox available within the MATLAB has a user-friendly graphical user interface (GUI) and allows to feed the input, set the target directly from a worksheet, build the ANN, train and test the network everything with the GUI without any coding. However, we caution that this approach could potentially make the students to think neural network as purely a black box modelling toolbox and will take away the opportunity to learn about the mathematical physics of the neural networks in the class.

For the exercise described, we obtained the experimental data from literature and the adsorption system was selected on purpose since it is one of the most studied unit operations in laboratories for a wide range of applications that include water capture, carbon capture, hydrogen and methane storage, removal of pollutants from wastewaters, air purification and the experimental data are available in plenty and the data can be easily obtained from the open literature. The neural network can be used to predict properties that may be of interest of the chemistry students like the melting point, boiling point of compounds and can be compared with the ones obtained from other methods (e.g., group contribution method). The topic of adsorption used in this exercise also fits within the scope of physical chemistry and thus can be easily implemented in chemistry classrooms.

Appendix A. Supplementary data

Supplementary material related to this article can be found in the online version, at doi:https://doi.org/10.1016/j.ecce.2021.04.003.

Declaration of Competing Interest

The authors report no declarations of interest.

References

ACM. ChEnected, [WWW Document]. Youtube. URL https://www.youtube.com/watch?v=puf7PRQbw feature=youtu.be (Accessed 12.20.20) 2019. Venkat Venkatasubramanian on Artificial Intelligence in Chemical Engineering. Anyoha, R., [WWW Document]. Spec. Ed. Artif. Intell. URL http://sites.hms.harvard.edu/flash/2017/history-artificial-intelligence (Accessed 11.5.20) 2017. The History of Artificial Intelligence – Science in the News. Azzam, M., Aramouni, N.A.K., Ahmad, M.N., Awad, M., Kwapinski, W., Zeaier, J., 2018. Dynamic optimization of dry reformer under catalyst sintering using neural networks. Energy Convers. Manage. 157, 146–156, http://dx.doi.org/10.1016/j.enconman.2017.11.089. Bagheri, M., Akbari, A., Mirbagheri, S.A., 2019. Advanced control of membrane fouling in filtration systems using artificial intelligence and machine learning techniques: a critical review. Process Saf. Environ. Prot. 123, 229–252. Beale, M.H., Hagan, M.T., Demuth, H.B., 2010. Neural network toolbox. User’s Guid. MathWorks 2, 77–81. Bezek, J.C., Hall, I.O., Clarke, L., 1993. Review of MR image segmentation techniques using pattern recognition. Med. Phys. Lancaster PA 20, 1033. Bi, W.L., Hosny, A., Schabath, M.B., Giger, M.L., Birkbak, N.J., Mehtas, A., Allison, T., Arnaout, O., Abboss, C., Dunn, I.F., 2019. Artificial intelligence in cancer imaging: clinical challenges and applications. CA Cancer J. Clin. 69, 127–157. Bishop, C.M., 1995. Neural Networks for Pattern Recognition. Oxford university press. Collins, A., Bobrow, D.G., 2017. Representation and Understanding: Studies in Cognitive Science. Academic Press Inc. Da Silva, A.H., Miranda, E.A., 2013. Adsorption/desorption of organic acids onto different adsorbents for their recovery from fermentation broths. J. Chem. Eng. Data 58, 1454–1463, http://dx.doi.org/10.1021/je3008799. Damour, C., Benne, M., Grodin-Perez, B., Chablat, J.-P., 2010. Nonlinear predictive control based on artificial neural network model for industrial crystallization. J. Food Eng. 99, 225–231, http://dx.doi.org/10.1016/j.jfoodeng.2010.02.027. Demuth, H., Beale, M., Hagan, M., 1992. Neural Network Toolbox. Use With MATLAB. MathWorks Inc. 2000. Deshpande, D., Pasapanodya, J.G., Mpagama, S.G., Bendet, P., Srivistava, S., Koeuth, T., Lee, P.S., Bhavnani, S.M., Ambrose, P.G., Thwaites, G., 2018. Levofloxacin pharmacokinetics/pharmacodynamics, dosing, susceptibility breakpoints, and artificial intelligence in the treatment of multidrug-resistant tuberculosis. Clin. Infect. Dis. 67, S293–S302. Dougherty, G., 2012. Pattern Recognition and Classification: an Introduction. Springer Science & Business Media. Fausett, L.V., 2006. Fundamentals of Neural Networks: Architectures, Algorithms and Applications. Pearson Education, India. Fleming, N., 2018. How artificial intelligence is changing drug discovery. Nature 557, 555–555. Freundlich, H.M.F., 1906. Over the adsorption in solution. J. Phys. Chem. 57, 1106–1107. Gammel, R.K., Rasmussen, Å.C., 2017. Estimation of melting temperature of molecular cocystals using artificial neural network model. Cryst. Growth Des. 17, 175–182, http://dx.doi.org/10.1016/j.cryseng.2014.04.030.
