TU Delft

Delft University of Technology

Solving vapor-liquid flash problems using artificial neural networks

Poort, Jonah P.; Ramdin, Mahinder; van Kranendonk, Jan; Vlugt, Thijs J.H.

DOI
10.1016/j.fluid.2019.02.023

Publication date
2019

Document Version
Accepted author manuscript

Published in
Fluid Phase Equilibria

Citation (APA)
Poort, J. P., Ramdin, M., van Kranendonk, J., & Vlugt, T. J. H. (2019). Solving vapor-liquid flash problems using artificial neural networks. *Fluid Phase Equilibria*, *490*, 39-47. https://doi.org/10.1016/j.fluid.2019.02.023

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.
For technical reasons the number of authors shown on this cover page is limited to a maximum of 10.

Solving vapor-liquid flash problems using artificial neural networks

Jonah P. Poort[a,b], Mahinder Ramdin[a], Jan van Kranendonk[b], Thijs J.H. Vlugt[a,*]

[a] *Engineering Thermodynamics, Process & Energy Department, Faculty of Mechanical, Maritime and Materials Engineering, Delft University of Technology, Leeghwaterstraat 39, 2628CB Delft, The Netherlands*

[b] *ZEF B.V., Leeghwaterstraat 39, 2628CB Delft, The Netherlands*

Abstract

Vapor-liquid phase equilibrium —flash— calculations largely contribute to the total computation time of many process simulations. As a result, process simulations, especially dynamic ones, are limited in the amount of detail that can be included due to simulation time restrictions. In this work, artificial neural networks were investigated as a potentially faster alternative to conventional flash calculation methods. The aim of this study is to extend existing applications of neural networks to fluid phase equilibrium problems by investigating both phase stability and property predictions. Multiple flash types are considered. Classification neural networks were used to determine phase stability, and regression networks were used to make predictions of thermodynamic properties. In addition to well established flash-types such as the pressure-temperature (PT), and pressure-entropy (PS) flashes, neural networks were used to develop two concept flashes: an entropy-volume (SV), and an enthalpy-volume (HV) flash. All neural networks were trained on, and compared to, data generated using the PT-flash from the Thermodynamics for Engineering Applications (TEA) property calculator. Training data was generated for binary water-methanol mixtures over a wide range of pressures and temperatures. Overall phase classification accuracy scores of around 97% were achieved. R^2 scores of property predictions were in the general order of 0.95 and higher. The artificial neural networks showed speed

*Corresponding author

Email address: t.j.h.vlugt@tudelft.nl (Thijs J.H. Vlugt)

© 2019 Manuscript version made available under CC-BY-NC-ND 4.0 license https://creativecommons.org/licenses/by-nc-nd/4.0/

improvements over TEA of up to 35 times for phase classification, and 15 times for property predictions.

Keywords: Vapor-Liquid Equilibria, Phase Stability, Flash Calculations, Artificial Neural Networks

1. Introduction

Process simulation is an indispensable tool in the design of process equipment, and investigating the viability of new, or optimizing the performance of current, process designs [1]. In addition, dynamic process simulation has become especially important for applications such as process control, simulation of start-up and shut-down behavior, and the analysis of inherently dynamic systems [2–5]. At the basis of any type of process simulation model are fluid property calculations [6], which require knowledge on the phase stability of a fluid. For an unstable mixture, computation of the vapor-liquid equilibrium compositions is performed; a so-called *flash* problem [1, 7–9]. Phase stability is conventionally determined by testing whether a state can be found which has a lower value of the relevant thermodynamic state function [7, 10–16]. Equilibrium calculation algorithms determine the two-phase compostions by iteratively solving a system of non-linear equations, or by minimization of a thermodynamic state function [17–26].

Executing a single flash calculation usually does not require long computational time, around one tenth of a milisecond for a single calculation — as was determined in this study. However, when incorporated in a (dynamic) process simulation, flash calculations are executed a large number of times, and significantly contribute to the required CPU time, up to 70% in some cases [27]. This can severely limit the amount of detail that can be considered within a single simulation. There is therefore an urgent need for faster alternatives to the conventional methods used for solving flash problems.

Artificial Neural Networks (ANN) are a group of computational methods that have attracted interest for a multitude of applications [28–34]. ANN are inspired by the biological networks of neurons found in animal brains, and are capable of learning complex non-linear relationships or structures within large sets of data by a process of repeated exposure and adaptation of the networks inner parameters to the data, referred to as training [35, 36]. One of the major benefits of ANN is that once it is properly trained, computations are very fast, as only matrix multiplication and addition is required, and at

most the execution of an exponential function. Moreover, ANN can process large numbers of input combinations in parallel [36].

ANN have been used extensively in the field of fluid phase equilibria, e.g. in the property prediction of hydrocarbons [37–39], refrigerants [40–44], petroleum fluids [27, 45–47], ionic liquids [48–52], alcohols [53, 54] and even food [55, 56]. In almost all cases the fluids studied were either pure compounds or fixed composition mixtures. In addition, ANN have been used to determine phase stability [27, 57] and vapor/liquid-liquid equilibria [58–61] of binary systems for a number of mixtures. In these studies, the number of compositions considered was often limited, and only Refs. [57] and [27] considered mixtures within the entire range of possible compositions. For a concise overview of current and future applications of machine learning in molecular science, the reader is referred to Ref. [62].

In this study, ANN were investigated as a potential alternative to conventional equation of state (EOS) based flash algorithms, including both phase stability analysis, and equilibrium and property calculations. The focus of this study was on binary mixtures of water and methanol of variable composition. The following flash types were considered: the pressure-temperature PT-flash, the pressure-entropy PS-flash, the entropy-volume SV-flash, and the enthalpy-volume HV-flash. To the best of our knowledge, the SV and HV flashes have no well-established implementation of a solution algorithm yet. The neural networks were trained on example data of binary mixtures of water and methanol generated using the Thermodynamics for Engineering Applications (TEA) property calculator [63] of the CAPE-OPEN to CAPE-OPEN process simulator (COCO) [64].

This manuscript is organized as follows. The next section will briefly cover the basic principles behind conventional flash calculations, as well as relevant concepts related to ANN. The subsequent section will outline the approach used in solving the flash problem, and provide details on the methods used to train the neural networks. Next, the accuracy scores and speed improvements will be presented and discussed. Finally, conclusions regarding the current investigation are provided. The current work shows that the ANN methods result in considerable speed improvements, while still achieving overall high predictive accuracy.

2. Theory

2.1. Flash Calculations

In general, a complete flash calculation can be divided into three separate parts: (1) phase stability analysis; (2) equilibrium calculations; and (3) property calculations. In the first part, equation of state (EOS) [65–67] or activity coefficient (AC) [68–70] models are used to determine whether a mixture remains a stable single phase (vapor or liquid), or splits into a state of vapor-liquid equilibrium [7]. This is done by investigating whether a two-phase composition exists which has a lower value of the state function than the single phase starting composition [7, 10–16]. If the system splits into two phases, the same EOS or AC models are used to calculate the equilibrium compositions of both phases until both the balance equations (mass, component, heat) and equilibrium conditions (equal temperature, pressure, and chemical potential/fugacity) are satisfied [17–26]. Once the equilibrium compositions are known, the EOS or AC models are used to calculate relevant properties [7–9]. If the stability analysis finds that the mixture is a stable single phase, the equilibrium calculations can be skipped and EOS or AC models are directly used to determine the properties. For a comprehensive explanation of the theory behind flash problems, the reader is referred to Ref. [7].

All fluid property calculations start with a specification of the mixture composition (n_i) as well as the values of two thermodynamic properties, which (according to Duhem's theorem [8]) fix the phase of the system, as well as the values of all other thermodynamic properties. Relevant thermodynamic properties are pressure (P), volume (V), temperature (T), entropy (S), enthalpy (H), internal energy (U), Gibbs free energy (G), and Helmholtz free energy (A). The two properties that are specified determine the type of the flash calculation, and practical algorithms exist for many different flash types [17, 18, 22, 23, 26, 71].

This work focusses on the application of artificial neural networks to four different flash types: the PT-, PS-, SV-, and HV-flash. The first two types have widely established algorithms [17, 18], and the third has a known algorithm [21], but is rarely implemented. To the best of our knowledge, for the last type no clearly defined algorithm is available.

2.2. Artificial Neural Networks

An artificial neural network (ANN) is made up of a network of connected nodes called neurons. All neurons are divided into mutually exclusive layers. In the first layer, called the input layer, the inputs to the network are defined. The last layer, the output layer, returns the output values of the network corresponding to the specified inputs. All layers in between the input and output layer are called hidden layers [35, 72]. Figure 1 shows a schematic representation of a neural network with three input neurons, two output neurons, and one hidden layer with four hidden neurons.

Each neuron in a network has a certain internal value called its activation. This value is passed on as a "signal" from one neuron to the next. In most ANN, every neuron in one layer is directionally connected to every neuron in the next layer. The neuron receives signals from the neurons in the layer directly before it, and sends signals to all neurons in the layer directly after it. Every connection between neuron i in one layer and neuron j in the next layer has a corresponding weight w_{ij}. The value of a weight indicates the connections "strength" and determines the magnitude of the signal that is propagated [35, 72]. Weights can be positive, having a stimulating effect, or negative, having an inhibiting effect.

All signals received by a neuron are added together and an *activation function* is applied to determine the value of the activation of the neuron. An additional term, called a bias (b_j), is added to the total signal, which makes it possible to shift the value of the activation, considerably improving the modelling capabilities of the network. In mathematical terms, signal propagation can be expressed as:

$$A_j = f_{\mathrm{a}}(b_j + \sum_{i=1}^{n} A_i w_{ij}) \tag{1}$$

Here, A_j is the activation of the j^{th} neuron in one layer, b_j is the bias term corresponding to neuron j, A_i is the activation of the i^{th} neuron in the layer before neuron j, w_{ij} is the weight value of the connection between neurons i and j, n is the total number of neurons in the layer before neuron j, and f_{a} is the activation function. Often used activation functions include the sigmoid, tanh, and Rectified Linear Unit functions [35], but many others exist [73, 74].

The most basic type of ANN is the Multi-Layer Perceptron (MLP), the type pictured in Figure 1, in which every neuron in one layer is forwardly connected to every neuron in the layer directly after it. The MLP can be

applied to many types of problems, including *classification* and *regression* [35]. In the case of regression applications, the simple one-layer MLP, with enough hidden neurons, has been proven to be a universal approximator of any *multi-variate* and *continuous* function [75–77].

Neural networks can learn complex input-output relations of non-linear problems. This is done in a process called *training*, in which the weights and biases of the network are adjusted to accurately represent the relation between a large number of example inputs and outputs. During training, the ANN is made to predict outputs based on example inputs, and the predictions made by the ANN are compared to corresponding correct example outputs. Based on the difference between the ANN predictions and target values, the network is given a performance (error) score, called its loss; determined by a loss function. As the loss is a function of the weights and biases of the ANN, an optimization algorithm can be used to find new values of the weights and biases that lead to a decrease of the loss value. After these new values are found, the weight and biases are updated accordingly, and the scoring-and-update process is repeated until either a maximum number of iterations (epochs) is exceeded, or a desired accuracy is achieved. In general, when more training data is available, a neural network can be trained to a higher accuracy [35].

3. Methodology

3.1. Solution Approach

The conventional flash problem was approached in a similar vein as it currently is, replacing all EOS based models and algorithms with ANN. Phase stability was determined using classification networks, and both equilibrium calculations and property value predictions were performed using regression networks. All applications used MLP networks.

The following thermodynamic properties were included in the current work: pressure (P), volume (V), temperature (T), entropy (S), enthalpy (H), internal energy (U), Gibbs free energy (G), Helmholtz free energy (A), and the two-phase liquid composition (x_i) and vapor composition (y_i). The two-phase compositions were calculated by first predicting values of the vapor fraction of each component (β_i), defined as:

$$\beta_i = \frac{n_i^{\mathrm{V}}}{n_i^{\mathrm{L}}} \tag{2}$$

Here, n_i is the number of moles of component i in [mol], and β_i is the vapor fraction of component i in [mol/mol]. The values of β_i were used to calculate the two-phase compositions x_i and y_i by the following equations: $n_i^{\mathrm{V}} = \beta_i n_i^{\mathrm{F}}$; $n_i^{\mathrm{L}} = n_i^{\mathrm{F}} - n_i^{\mathrm{V}}$; $y_i = \dfrac{n_i^{\mathrm{V}}}{\sum_{i=1}^{c} n_i^{\mathrm{V}}}$; and $x_i = \dfrac{n_i^{\mathrm{L}}}{\sum_{i=1}^{c} n_i^{\mathrm{L}}}$. Here, x_i is the mole fraction of component i in the liquid phase in [mol/mol], y_i is the mole fraction of component i in the vapor phase in [mol/mol], c is the total number of components in the mixture, and V, L, and F are superscripts indicating the vapor phase, liquid phase, and feed respectively. This method was used to keep the number of independent variables that had to be predicted to determine the two-phase compositions to a minimum, as well as to insure that the mass balance was satisfied.

It was decided to use a different neural network for each property prediction individually, instead of using one network to predict all properties at once. This approach was used for two main reasons: (1) it allowed for the addition and removal of properties from the overall algorithm when necessary, without having to retrain already existing networks; and (2) better insights on which properties were difficult to train a neural network on could be gained. As the error score of a network with multiple outputs is determined by the sum of the error scores of each individual output, a single output that scores poorly can severely skew the total error score, even when the other properties score well. This can make it difficult to distinguish between properties that are easily approximated by a neural network, and properties that are not.

To avoid discontinuities of property values that may occur across phase boundary transitions, it was decided to divide the entire range of interest in three separate regions: the pure vapor region, pure liquid region, and two-phase vapor-liquid equilibrium region. For every property, a different neural network was trained for each phase region, and the classification network was used to determine the appropriate network to employ.

3.2. Data

Data was generated for 101 different compositions (z_i) of binary water-methanol mixtures for 500 temperatures evenly distributed between 273 and 700 K, and 500 pressures logarithmically distributed between 10^4 and $3{\cdot}10^7$ Pa, amounting to a total of around 25,250,000 data points. For a small number of PTz input combinations, the flash algorithm failed to converge to a solution, so the exact total of data points was 25,249,692; 15,927,634

points in the vapor region; 8,905,225 in the liquid region; and 416,833 in the two-phase region. The comparatively low number of data points in the two-phase region resulted from the small size of the water-methanol PT two-phase region. This is reflected in Figure 2, which shows a PT phase diagram that was part of the training data used to train the PT-flash classifier. As can be seen, the two-phase region (light-green) has a much smaller area than the pure liquid region (blue) and the pure vapor region (red).

The PT-flash algorithm incorporated in TEA was used to generate all data points. The algorithm uses a modified version of the Boston and Britt "inside-out" flash algorithm [20, 78]. The Peng-Robinson (PR) EOS [66] was used for all calculations (phase stability, equilibrium calculations, and property calculations). It is important to note that advanced EOSs like PC-SAFT or CPA are better than PR to model polar systems (e.g., water and methanol) [7]. The PR EOS was selected here due to its simplicity and robustness, but the working principle of the ANN method is independent of the choice of the EOS.

In total 70% of the data was used for training the neural networks, 15% was used for validation during the training process to prevent overfitting [35], and 15% was set aside for testing purposes. Before training, all data sets were scaled to lie in the interval [0, 1] using the following equation:

$$d'_i = \frac{d_i - \max(\boldsymbol{d})}{\max(\boldsymbol{d}) - \min(\boldsymbol{d})} \tag{3}$$

Here, d'_i is the scaled value of data point d_i, the i^{th} entry of data set $\boldsymbol{d}$. Scaling was applied to prevent immediate saturation of the activation functions, and to equalize the scales of the different input properties.

3.3. Neural Network Training

Neural networks were trained using the Keras module for the Python programming language [79]. All networks were MLP networks with a single hidden layer. The classification networks were more prone to overfitting, so were constructed with only 5 hidden neurons. The regression networks were all constructed with 10 hidden neurons. All networks used the sigmoid activation function for the hidden layer. The classification networks used the softmax function as the activation function for the output layer, the regression networks had no activation function in the output layer (linear activation).

For training the classification networks, the output labels corresponding to each phase region were transformed into one-hot encodings, and trained using the categorical cross-entropy (CE) loss function [35]:

$$\mathrm{CE} = -\frac{1}{n}\sum_{i=1}^{n}[t_i \log o_i + (1 - t_i)\log(1 - o_i)] \tag{4}$$

Here, t_i is the target value of data instance i, o_i is the value predicted by the ANN on data instance i, and n is the total number of data instances.

The regression networks were trained using the mean squared error (MSE) loss function. The MSE is defined as:

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(t_i - o_i)^2 \tag{5}$$

For all networks, the Nadam algorithm was used to minimize the loss value [80]. The total training data set was processed by the training algorithm in batches of 5000-15000 randomly selected instances. An early-stopping monitor was implemented which ended the training process if no decrease in validation error of at least $1{\cdot}10^{-6}$ was achieved for more than 35 epochs. The maximum number of training epochs was set to 5000.

4. Results and Discussion

4.1. Predictive Accuracy

To score accuracy, new sets of test data were generated using TEA. The data was generated by randomly selecting PTz combinations within the range of interest specified in Section 3.2, and determining their phase stability and target property values. Randomly generated test instances were used instead of the original test data set, so that any biases that might have been introduced by the manner in which the training data was generated would become apparent. For each phase region, 100,000 points were generated.

The phase classification neural networks were scored based on a binary percentage accuracy measurement, defined as:

$$\mathrm{BA} = \frac{n_{\mathrm{cor}}}{n} \cdot 100\% \tag{6}$$

Here, n_{cor} is the number of correct predictions and n is the total number of predictions made.

The regression networks were scored based on their mean absolute error (MAE) and coefficient of deviation (R^2), defined as:

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|t_i - o_i| \tag{7}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n-1}(t_i - o_i)^2}{\sum_{i=1}^{n-1}(t_i - \mu_{\mathrm{test}})^2} \tag{8}$$

Here, t_i is the target output value of instance i, o_i is the predicted output value of instance i, and μ_{test} is the mean value of the test data set.

Table 1 shows the binary accuracy scores as percentages for each flash type. All flash types were scored on each phase region separately, as well on the total test data of all regions combined. As can be seen, all flash types show similar results: high accuracy for points in the liquid and vapor regions, especially for the SV- and HV-flash types, but markedly lower accuracy for points in the two-phase region, which have a significant impact on the overall accuracy scores. The lower accuracy scores for the two-phase regions are most likely due to the fact that less training data was available for the two-phase region.

Table 2 shows accuracy scores of the PT- PS- and SV-flash for a selection of properties in each phase region. As can be seen, for the PT-flash, accuracy scores are quite good in the liquid and vapor regions, with relatively low mean absolute errors for most properties, and values for the coefficient of deviation close to 1. Figure 3 shows a correlation plot of the neural network PT-flash predictions of the Gibbs free energy compared to the target values in the test data set. In the figure, points whose phase was incorrectly classified by the classification neural network that preceded the property prediction are shown as red crosses, the blue circles indicate correctly classified points. As can be seen, the neural network accurately predicts the target values.

As opposed to the vapor and liquid regions, the two-phase region accuracy scores are overall much lower. Except for the volume and the Gibbs free energy, all properties show an increase in the MAE that is in many cases around an order of magnitude larger than for the liquid or vapor region. These results can be explained by the fact that the PT two-phase region shows relatively steep gradients for many properties. These gradients can make it difficult for a neural network to accurately approximate the training data. Additionally, the fact that less training data was available for the

two-phase region also likely contributed to the lower accuracy scores. Table 2 further shows that the accuracy of the vapor fraction of water and liquid phase mole fraction of water are considerably worse than all other properties. For the vapor fraction of water this is presumably due to errors in the phase classification that precedes property value predictions. As the value of the vapor fraction in the liquid region is always 0, and in the vapor region always 1, a misclassification in the two-phase region will lead to disproportionately large errors in the value of the vapor fraction. Accuracy scores for the liquid phase mole fraction of water are worse than the vapor fraction of water for an additional reason: as the liquid mole fraction is derived from both the water and methanol vapor fractions, predictive errors in both vapor fractions will combine to result in even larger errors in the mole fraction. Accuracy scores of the methanol vapor fraction and liquid mole fraction, as well as vapor mole fractions showed similar results (data not shown).

The PS-flash shows vapor and liquid region results that are comparable to the results of the PT-flash, but the PS-flash is much more accurate in the two-phase region. In particular, the vapor fraction of water shows a much lower MAE and higher R^2 score, likely due to the fact that in the PS two-phase region, property values do not show the steep gradients they show in the PT two-phase region. The accuracy of the liquid phase mole fraction of water is still low, but better than for the PT-flash. Figure 4 shows a similar plot as Figure 3 in which the neural network PS-flash predictions for enthalpy are compared to their target values in the test data set. Again, it can be seen that the neural network has very high predictive performance.

Table 2 likewise shows similar trends in accuracy scores of the SV-flash as the PT- and PS-flash. However, there is one main difference: pressure predictions show very large errors, likely due to the large value range of pressure, and, in the liquid region, due to very steep gradients similar to the PT two-phase region, but more severe. Figure 5 shows that the high errors for the two-phase region pressure predictions are almost entirely due to errors in the preceding phase classification network. As pressure predictions in the liquid region are highly inaccurate, any point in the two-phase region that is incorrectly classified as being in the liquid region will contribute significantly to the total error score of the two-phase region. In fact, when excluding incorrect phase classifications, the SV-flash two-phase pressure predictions showed an MAE value of $6.8{\cdot}10^4$ Pa, compared to a value of $1.8{\cdot}10^6$ Pa when incorrect classification are included. Thus, around 96% of of the value of the MAE in the two-phase region is due to points that were incorrectly

classified, mostly as liquid. For comparison, in the vapor region, only 0.01% of the MAE value can be contributed to incorrect classifications. Most other properties also show much lower error contributions from misclassification. For example, Figure 6 compares temperature values predicted using the SV-flash neural network to their target values. As can be seen, it shows only a few points that were incorrectly classified.

Results for the HV-flash were not reported as they were almost identical to the results for the SV-flash, even showing the same difficulty in predicting values for pressure in the liquid region.

4.2. Execution Time

Algorithm execution times were determined using the time module for the Python programming language. Using the module, the time just before the function call to the algorithm and just after the algorithm returned its output were recorded. The time before execution was subtracted from the time after execution to determine the total time elapsed. All time measurements were performed on a MSI GP62 Leopard laptop with a 64-bit Intel®Core™ i7-7700HQ 2.80GHz processor running the Windows 10 OS, and 8.00 GB of installed ram. All algorithms were executed using the same amount of CPU space.

Execution times were determined for three cases: the conventional algorithm used by TEA [20], a phase classification neural network only, and a property prediction neural network that was preceded by the classification network. As all networks had the same number of neurons and used the same activation functions, only one property of one flash type was considered.

One of the major advantages of neural networks is that they can process multiple inputs in parallel, for this reason it was interesting to investigate how execution time scaled with the total number of flashes to execute. The number of flashes to execute were: 10, 10^2, 10^3, 10^4, 10^5, 10^6, and 10^7. All cases were repeated a set number of times; their averages are reported here.

Figure 7 shows a log-log plot of the total average execution time versus the number of flashes that had to be executed for each of the three cases. As can be seen, from 10^4 and onwards, execution time scales roughly linearly with an increase in the number of flashes to execute for all cases. Below 10^4 calculations, the execution times of the neural networks stop decreasing linearly. As a result, at low number of flashes to execute, the neural networks show lower, but still significant, speed improvements over TEA. Speed improvements were determined by dividing the execution time of the TEA

algorithm by the execution time of the neural network algorithms. At low numbers of flashes to execute (10-10^3), speed improvements are around 6-25 times for the classification network only, and 2-10 times for property prediction preceded by phase classification. At higher numbers of flashes to execute (10^4-10^6), the neural networks show even better improvements over TEA, up to 35 times for classification, and 15 times for property prediction. For the highest number of flashes to execute (10^7), improvements reach values of 90 and 35 times, respectively. The speed improvements of 35 and 15 times correspond to percentage decreses of around 97% and 92%, respectively. These results are comparable to the maximum speedup of 90% achieved by [27]. Although it should be mentioned that the speedup in [27] was measured based on a decrease in the number of Rachford-Rice iterations, and not explicit execution time.

The speed improvements are due to two reasons. Firstly, neural networks do not require any iterative process or the solution to a system of complex equations, making them relatively simple and fast to execute. Secondly, due to their architecture, neural networks can execute multiple flash calculations at once, while the TEA algorithm can only handle one flash at a time. As the results showed, the classification plus prediction case is considerably slower than the classification only case. This is mainly due to the fact that two neural networks had to be executed, as opposed to only one for the classification only case. In addition, auxiliary functions (scaling, splitting and formatting arrays, rescaling, etc.) contributed an additional 5-10% to the execution time of the classification plus prediction case.

5. Conclusion

In this work, an algorithm was developed that uses artificial neural networks to replace the complex equations and iterative processes present in conventional flash algorithms. This leads to faster flash calculations. A total of four different flash types were considered for binary mixtures of water and methanol of variable composition: the PT-, PS-, SV-, and HV-flash. The classic flash problem was divided into two separate problems: one of classifying the phase region (vapor, liquid, or two-phase equilibrium), and one of predicting property values of the mixture. Phase classification showed overall quite high accuracy scores (around 97%), although classification accuracy of the two-phase region was considerably lower than for the pure liquid and vapor phase regions. Property predictions showed good accuracy for many

properties ($R^2 > 0.95$), but again predictions of values in the two-phase region showed much higher errors. Lower two-phase region accuracy scores were likely due to less training data being available for points in this region, and steep property value gradients that occurred for the PT-flash. Furthermore, execution time improvements of up to 35 times for phase classification, and 15 times for property value predictions were achieved. The results from this study show that ANN can be used to rapidly and accurately predict phase stability and values of many different properties in the vapor and liquid regions, but still require improvements when it comes to the accuracy in the two-phase vapor-liquid equilibrium region. Finally, the results show that ANN can be used to develop algorithms for unconventional flash types such as the SV and HV-flash.

Acknowledgements

The authors thank Zero Emission Fuels (ZEF B.V.) for sponsoring and supporting this study. TJHV acknowledges NWO-CW (Chemical Sciences) for a VICI grant.

Table 1: Binary accuracy scores of of all flash types considered. Binary accuracy was defined as the percentage of the total number of predictions that were made that were correct. Separate accuracy scores are shown for the pure liquid region, pure vapor region, and two-phase region. The overall score over all phase regions together is also shown.

	Binary Accuracy [%]			
Flash type	Liquid	Vapor	Two-phase	Overall
PT	99.8	99.7	91.0	96.8
PS	99.7	99.7	93.4	97.6
SV	99.9	99.99	91.0	97.0
HV	100.0	99.997	90.1	96.7

Table 2: Accuracy scores of the PT-, PS-, and SV-flash for pressure (P), volume (V), temperature (T), entropy (S), enthalpy (H), and Gibbs free energy (G). Results are shown for the liquid, vapor, and two-phase regions. For the two-phase region, scores for the vapor fraction of water (β_{w}), and liquid composition of water (x_{w}) are also shown. The accuracy scores shown are the mean absolute error (MAE) and the coefficient of deviation (R^2).

	Liquid region					
	***PT*-flash**		***PS*-flash**		***SV*-flash**	
Property	MAE	R^2	MAE	R^2	MAE	R^2
P	-	-	-	-	$1.6{\cdot}10^{7}$	-6.2
V	$7.7{\cdot}10^{-7}$	$9.790{\cdot}10^{-1}$	$6.1{\cdot}10^{-7}$	$9.797{\cdot}10^{-1}$	-	-
T	-	-	1.2	$9.994{\cdot}10^{-1}$	2.0	$9.976{\cdot}10^{-1}$
S	$4.0{\cdot}10^{-1}$	$9.991{\cdot}10^{-1}$	-	-	-	-
H	$2.0{\cdot}10^{2}$	$9.987{\cdot}10^{-1}$	$1.3{\cdot}10^{2}$	$9.998{\cdot}10^{-1}$	$3.6{\cdot}10^{2}$	$9.986{\cdot}10^{-1}$
G	$1.0{\cdot}10^{2}$	$9.997{\cdot}10^{-1}$	$1.3{\cdot}10^{2}$	$9.995{\cdot}10^{-1}$	$2.8{\cdot}10^{2}$	$9.974{\cdot}10^{-1}$
	Vapor region					
	***PT*-flash**		***PS*-flash**		***SV*-flash**	
Property	MAE	R^2	MAE	R^2	MAE	R^2
P	-	-	-	-	$6.0{\cdot}10^{5}$	$9.785{\cdot}10^{-1}$
V	$2.1{\cdot}10^{-5}$	$9.999{\cdot}10^{-1}$	$4.4{\cdot}10^{-5}$	$9.993{\cdot}10^{-1}$	-	-
T	-	-	3.0	$9.944{\cdot}10^{-1}$	$9.0{\cdot}10^{-1}$	$9.995{\cdot}10^{-1}$
S	$4.3{\cdot}10^{-1}$	$9.960{\cdot}10^{-1}$	-	-	-	-
H	$2.8{\cdot}10^{2}$	$9.866{\cdot}10^{-1}$	$1.3{\cdot}10^{2}$	$9.983{\cdot}10^{-1}$	$2.0{\cdot}10^{2}$	$9.961{\cdot}10^{-1}$
G	$1.5{\cdot}10^{2}$	$9.986{\cdot}10^{-1}$	$1.9{\cdot}10^{2}$	$9.976{\cdot}10^{-1}$	$2.6{\cdot}10^{2}$	$9.953{\cdot}10^{-1}$
	Two-phase region					
	***PT*-flash**		***PS*-flash**		***SV*-flash**	
Property	MAE	R^2	MAE	R^2	MAE	R^2
P	-	-	-	-	$1.8{\cdot}10^{6}$	-2.513
V	$2.1{\cdot}10^{-5}$	$9.664{\cdot}10^{-1}$	$2.6{\cdot}10^{-5}$	$9.995{\cdot}10^{-1}$	-	-
T	-	-	1.2	$9.972{\cdot}10^{-1}$	2.7	$9.732{\cdot}10^{-1}$
S	2.7	$9.479{\cdot}10^{-1}$	-	-	-	-
H	$1.3{\cdot}10^{3}$	$9.431{\cdot}10^{-1}$	$1.2{\cdot}10^{2}$	$9.996{\cdot}10^{-1}$	$1.4{\cdot}10^{2}$	$9.989{\cdot}10^{-1}$
G	$5.5{\cdot}10^{1}$	$9.995{\cdot}10^{-1}$	$8.8{\cdot}10^{1}$	$9.987{\cdot}10^{-1}$	$2.8{\cdot}10^{2}$	$9.304{\cdot}10^{-1}$
β_{w}	$5.0{\cdot}10^{-2}$	$8.821{\cdot}10^{-1}$	$1.5{\cdot}10^{-2}$	$9.887{\cdot}10^{-1}$	$1.6{\cdot}10^{-2}$	$9.852{\cdot}10^{-1}$
x_{w}	$7.1{\cdot}10^{-2}$	$9.823{\cdot}10^{-2}$	$3.6{\cdot}10^{-2}$	$4.644{\cdot}10^{-1}$	$3.0{\cdot}10^{-2}$	$5.839{\cdot}10^{-1}$

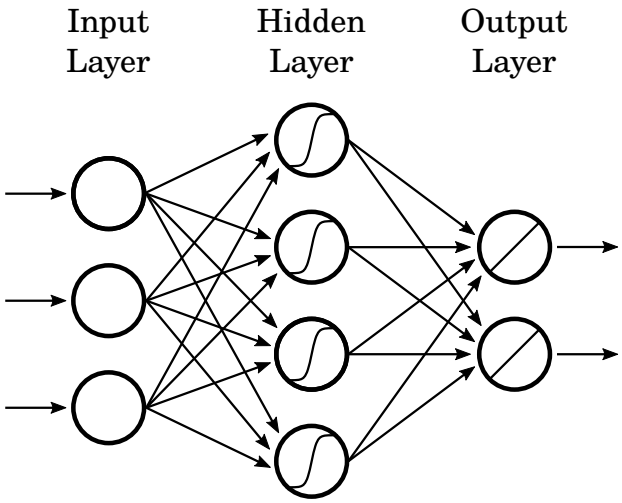


Figure 1: Schematic representation of a Multi-Layer Perceptron type artificial neural network [35] with three neurons in the input layer, two neurons in the output layer, and one hidden layer with four hidden neurons. Every neuron in one layer is connected to all neurons in the next layer. Information is propagated only in the forwards direction, from the input layer, through the hidden layer, to the output layer.

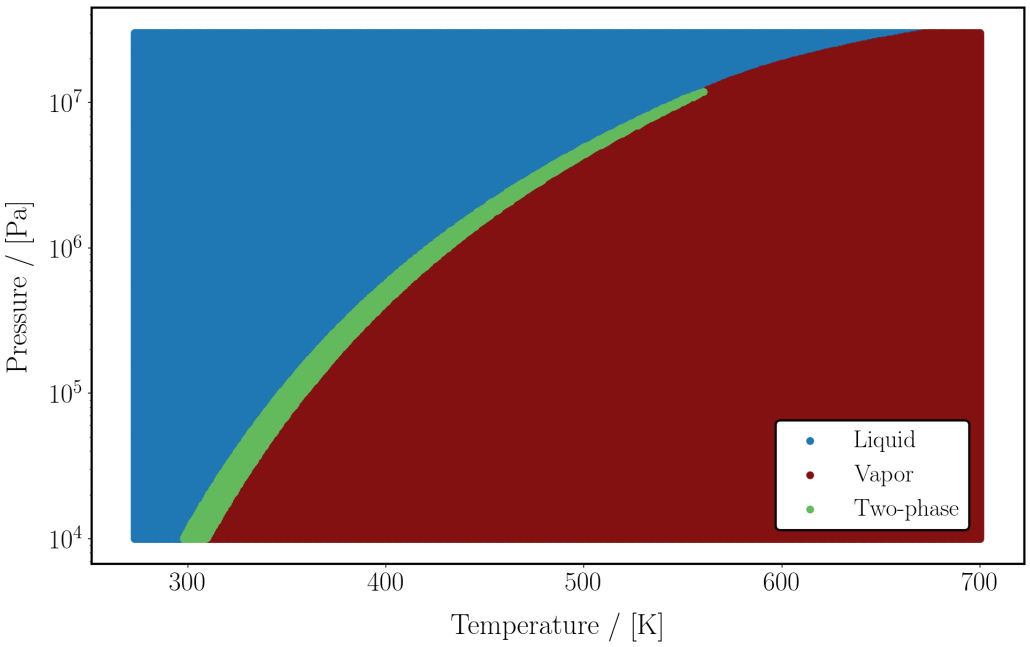


Figure 2: Graphical representation of a part of the phase region data which was generated in order to train and validate the PT phase classification neural network. The liquid region is shown in blue, the vapor region in red, and the two-phase region in light-green. Data is shown for a 50-50 mol% water-methanol mixture. The actual neural network is trained on 101 such data sets. The data was generated using the PT-flash algorithm from the Thermodynamics for Engineering Application property calculator [20, 63].

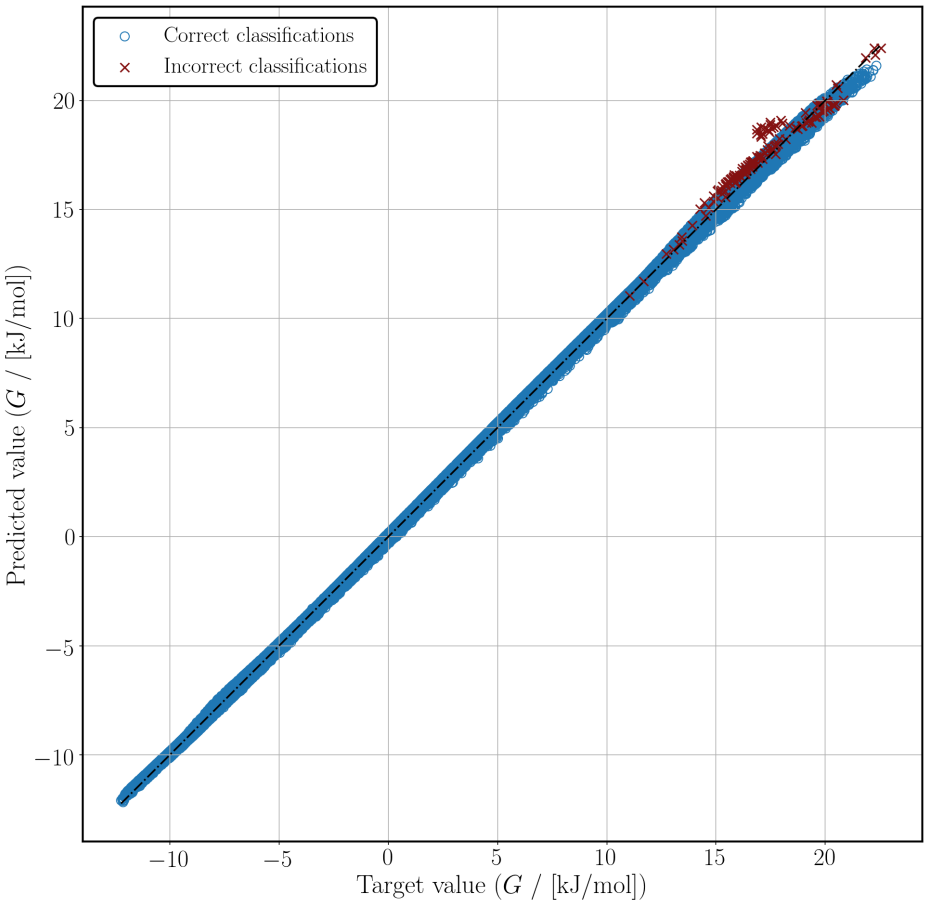


Figure 3: Target Gibbs free energy (G) values versus predicted Gibbs free energy values for the PT-flash liquid region. Points that were correctly classified as being in the two-phase region are shown as blue circles, while incorrectly classified points are shown as red crosses. As can be seen, the neural network accurately predict the target value ($R^2 = 0.9997$), and incorrect phase classification contribute little to the algorithms predictive error.

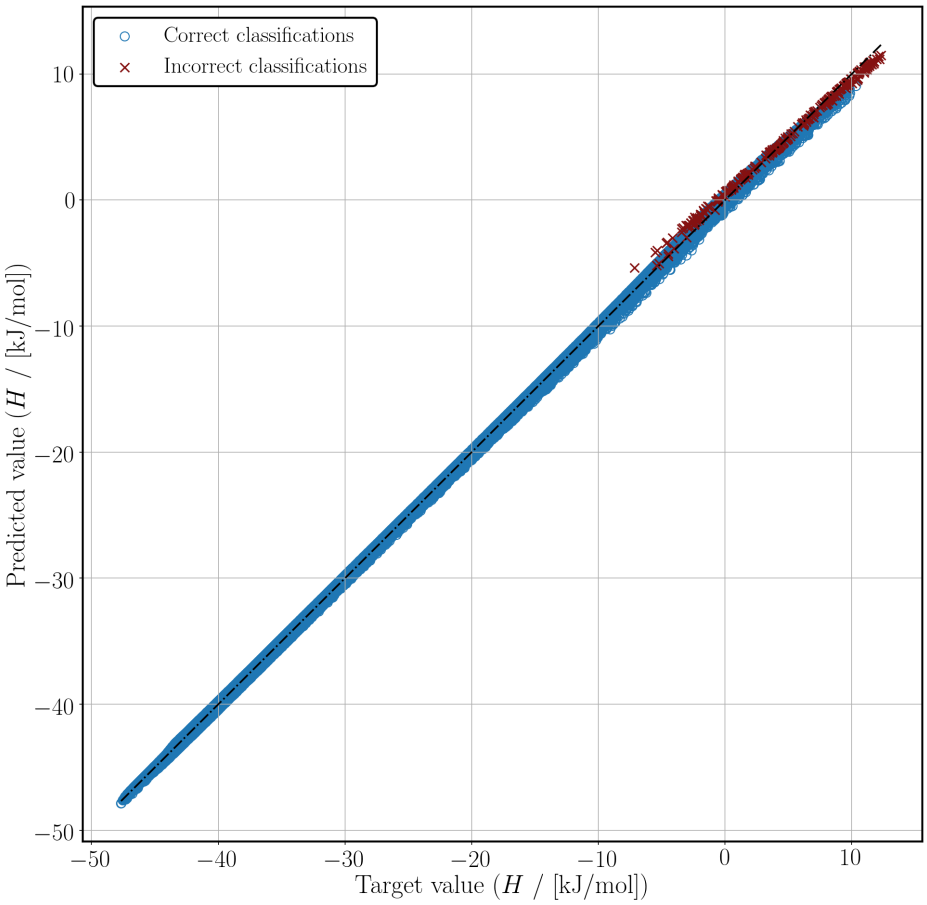


Figure 4: Target enthalpy (H) values versus predicted enthalpy values for the PS-flash liquid region. Points that were correctly classified as being in the two-phase region are shown as blue circles, while incorrectly classified points are shown as red crosses. As can be seen, the neural network accurately predict the target value ($R^2 = 0.9998$), and incorrect phase classification contribute little to the algorithms predictive error.

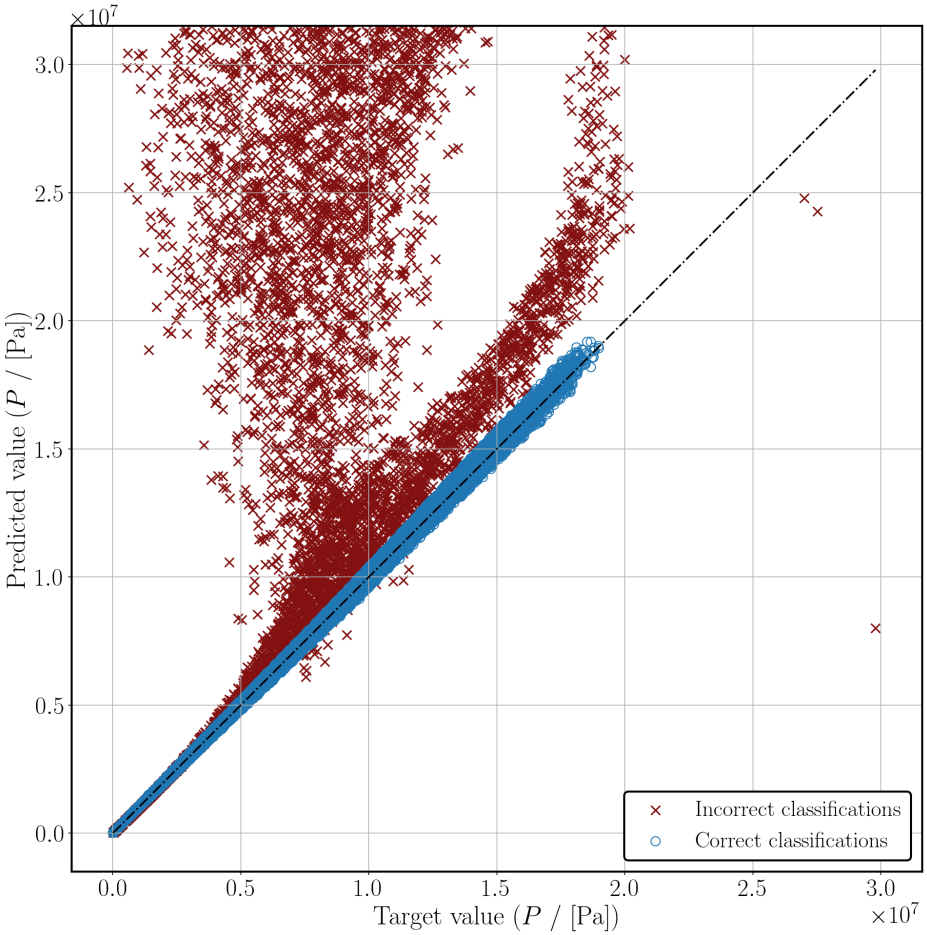


Figure 5: Target pressure (P) values versus predicted pressure values for the SV-flash two-phase region. Points that were correctly classified as being in the two-phase region are shown as blue circles, while incorrectly classified points are shown as red crosses. The incorrect classifications make up only around 9% of the total number of predictions made, however, they contribute to around 96% of the total value of the mean absolute error. In addition, when only looking at the correctly classified points, the correlation is very good ($R^2 = 0.9993$), but when taking into account all points, including incorrect classifications, overall accuracy is much worse (R^2 = -2.51).

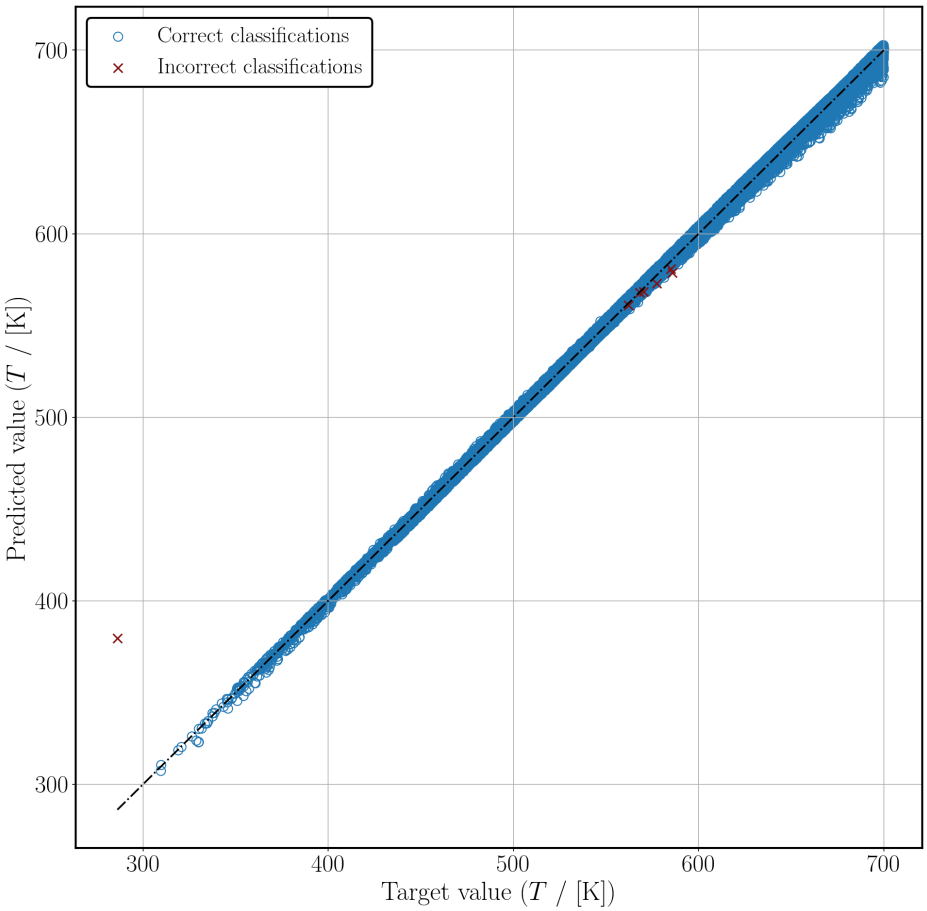


Figure 6: Target temperature (T) values versus predicted temperature values for the SV-flash vapor region. Points that were correctly classified as being in the two-phase region are shown as blue circles, while incorrectly classified points are shown as red crosses. As can be seen, the neural network accurately predict the target value ($R^2 = 0.9995$), and incorrect phase classification contribute little to the algorithms predictive error.

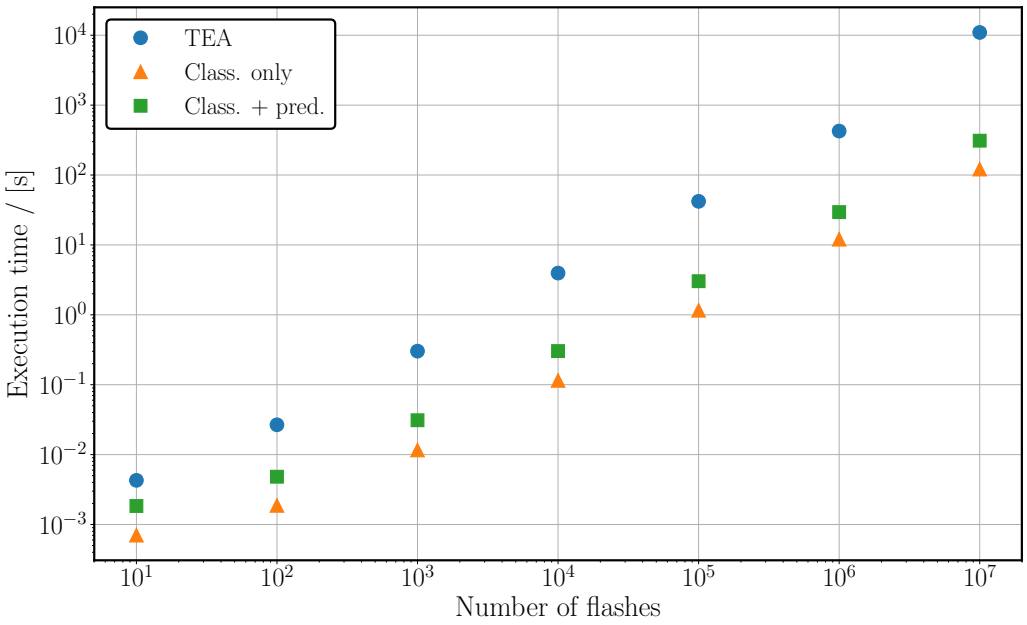


Figure 7: Average execution times versus the number of flashes to execute. A comparison is made between the conventional algorithm from the TEA property calculator [20, 63] (blue circles), the classification neural network only (orange triangles), and a property prediction which was preceded by the phase classification network (green squares). All time measurements were performed on a MSI GP62 Leopard laptop with a 64-bit Intel®Core™ i7-7700HQ 2.80GHz processor running the Windows 10 OS, and 8.00 GB of installed ram. All algorithms were executed using the same amount of CPU space. Training times were not included.

[1] J. Seader, E. J. Henley, D. K. Roper, Separation Process Principles, 4th Edition, John Wiley & Sons, Hoboken, 2016.

[2] L. Naess, A. Mjaavatten, J. O. Li, Using dynamic process simulation from conception to normal operation of process plants, Comput. Chem. Eng. 17 (5-6) (1993) 585–600.

[3] AspenTech Documentation Team, Aspen HYSYS Dynamic Modeling Guide, Tech. rep. (2006).

[4] W. L. Luyben, Use of dynamic simulation for reactor safety analysis, Comput. Chem. Eng. 40 (2012) 97–109.

[5] F. Alobaid, R. Starkloff, S. Pfeiffer, K. Karner, B. Epple, H. G. Kim, A comparative study of different dynamic process simulation codes for combined cycle power plants-Part B: Start-up procedure, Fuel 153 (2015) 707–716.

[6] G. M. Kontogeorgis, R. Gani, Computer-Aided Property Estimation for Process and Product Design, 1st Edition, Elsevier B.V., Amsterdam, 2004.

[7] M. L. Michelsen, J. M. Mollerup, Thermodynamic Models: Fundamental & Computational Aspects, 2nd Edition, Tie-Line Publications, Holte, 2007.

[8] J. Smith, H. Van Ness, M. Abbott, Introduction to Chemical Engineering Thermodynamics, sixth Edition, McGraw-Hill Higher Education, New York, 2003.

[9] S. M. Walas, Phase Equilibria in Chemical Engineering, 1st Edition, Butterworth Publishers, Boston, 1985.

[10] M. L. Michelsen, The isothermal flash problem. Part I. Stability, Fluid Phase Equilib. 9 (1) (1982) 1–19.

[11] H. Gecegormez, Y. Demirel, Phase stability analysis using interval Newton method with NRTL model, Fluid Phase Equilib. 237 (1-2) (2005) 48–58.

[12] J. Mikyška, A. Firoozabadi, Investigation of mixture stability at given volume, temperature, and number of moles, Fluid Phase Equilib. 321 (2012) 1–9.

[13] M. Castier, Helmholtz function-based global phase stability test and its link to the isothermal-isochoric flash problem, Fluid Phase Equilib. 379 (2014) 104–111.

[14] D. V. Nichita, Fast and robust phase stability testing at isothermal-isochoric conditions, Fluid Phase Equilib. 447 (2017) 107–124.

[15] T. Smejkal, J. Mikyška, Phase stability testing and phase equilibrium calculation at specified internal energy, volume, and moles, Fluid Phase Equilib. 431 (2017) 82–96.

[16] T. Smejkal, J. Mikyška, Unified presentation and comparison of various formulations of the phase stability and phase equilibrium calculation problems, Fluid Phase Equilib. 476 (2018) 61–88.

[17] M. L. Michelsen, The Isothermal Flash Problem. Part II. Phase-Split Calculation, Fluid Phase Equilib. 9 (1982) 21–40.

[18] M. L. Michelsen, Multiphase Isenthalpic and Isentropic Algorithms, Fluid Phase Equilib. 33 (1987) 13–27.

[19] C. H. Whitson, M. L. Michelsen, The negative flash, Fluid Phase Equilib. 53 (C) (1989) 51–71.

[20] V. S. Parekh, P. M. Mathias, Efficient Flash Calculations for Chemical Process Design Extension of the BostonBritt "Insideout" Flash Algorithm to Extreme Conditions and New Flash Types, Comput. Chem. Eng. 22 (10) (1998) 1371–1380.

[21] M. L. Michelsen, State function based flash specifications, Fluid Phase Equilib. 158 (1999) 617–626.

[22] M. Castier, Solution of the isochoric-isoenergetic flash problem by direct entropy maximization, Fluid Phase Equilib. 276 (1) (2009) 7–17.

[23] T. Jindrová, J. Mikyška, Fast and robust algorithm for calculation of two-phase equilibria at given volume, temperature, and moles, Fluid Phase Equilib. 353 (2013) 101–114.

[24] M. Petitfrere, D. V. Nichita, Robust and efficient Trust-Region based stability analysis and multiphase flash calculations, Fluid Phase Equilib. 362 (2014) 51–68.

[25] H. Fatoorehchi, H. Abolghasemi, R. Rach, A new parametric algorithm for isothermal fl ash calculations by the Adomian decomposition of Michaelis Menten type nonlinearities, Fluid Phase Equilib. 395 (2015) 44–50.

[26] M. Castier, R. Kanes, L. N. Véchot, Flash calculations with specified entropy and stagnation enthalpy, Fluid Phase Equilib. 408 (2016) 196–204.

[27] A. Kashinath, M. Szulczewski, A. Dogru, A fast algorithm for calculating isothermal phase behavior using machine learning, Fluid Phase Equilib. 465 (2018) 73–82.

[28] A. Statnikov, C. F. Aliferis, I. Tsamardinos, D. Hardin, S. Levy, A comprehensive evaluation of multicategory classification methods for microarray gene expression cancer diagnosis, Bioinformatics 21 (5) (2005) 631–643.

[29] E. Guresen, G. Kayakutlu, T. U. Daim, Using artificial neural network models in stock market index prediction, Expert Syst. Appl. 38 (8) (2011) 10389–10397.

[30] T. Sagara, M. Hagiwara, Natural language neural network and its application to question-answering system, Neurocomputing 142 (2014) 201–208.

[31] Y. Zhao, X. Du, G. Xia, L. Wu, A novel algorithm for wavelet neural networks with application to enhanced PID controller design, Neurocomputing 158 (2015) 257–267.

[32] U. Fiore, A. De Santis, F. Perla, P. Zanetti, F. Palmieri, Using generative adversarial networks for improving classification effectiveness in credit card fraud detection, Inf. Sci. 0 (2017) 1–8.

[33] J. Dou, C. Liu, B. Wang, Short-term Wind Power Forecasting Based on Convolutional Neural Networks, IOP Conf. Ser. Earth Environ. Sci. 170 (4) (2018) 042023.

[34] H. Li, P. Wang, M. You, C. Shen, Reading car license plates using deep neural networks, Image Vis. Comput. 72 (2018) 14–23.

[35] A. Géron, Hands-On Machine Learning with Scikit-Learn and TensorFlow, 1st Edition, O'Reilly Media, 2017.

[36] A. I. Galushkin, Neural Networks Theory, 1st Edition, Springer-Verlag, Berlin; Heidelberg, 2007.

[37] J. Homer, S. C. Generalis, J. H. Robson, Artificial neural networks for the prediction of liquid viscosity, density, heat of vaporization, boiling point and Pitzer's acentric factor. Part I. Hydrocarbons, Phys. Chem. Chem. Phys. 1 (17) (1999) 4075–4081.

[38] R. Haghbakhsh, H. Adib, P. Keshavarz, M. Koolivand, S. Keshtkari, Development of an artificial neural network model for the prediction of hydrocarbon density at high-pressure, high-temperature conditions, Thermochim. Acta 551 (2013) 124–130.

[39] M. Moosavi, N. Soltani, Prediction of hydrocarbon densities using an artificial neural network-group contribution method up to high temperatures and pressures, Thermochim. Acta 556 (2013) 89–96.

[40] A. Chouai, S. Laugier, D. Richon, Modeling of thermodynamic properties using neural networks, Fluid Phase Equilib. 199 (1-2) (2002) 53–62.

[41] S. Laugier, D. Richon, Use of artificial neural networks for calculating derived thermodynamic quantities from volumetric property data, Fluid Phase Equilib. 210 (2) (2003) 247–255.

[42] A. Sözen, M. Özalp, E. Arcakliolu, Calculation for the thermodynamic properties of an alternative refrigerant (R508b) using artificial neural network, Appl. Therm. Eng. 27 (2-3) (2007) 551–559.

[43] E. R. Mora, P. Carlos, F. Gonza, J. D. Dios, D. Ocampo, Thermodynamic properties of refrigerants using artificial neural networks, Int. J. Refrig. 6 (2014) 9–16.

[44] Á. Mulero, I. Cachadiña, J. Valderrama, Artificial neural network for the correlation and prediction of surface tension of refrigerants, Fluid Phase Equilib. 451 (2017) 60–67.

[45] M. Seifi, J. Abedi, An efficient and robust saturation pressure calculation algorithm for petroleum reservoir fluids using a neural network, Pet. Sci. Technol. 30 (22) (2012) 2329–2340.

[46] M. Al-Marhoun, E. Osman, Using Artificial Neural Networks to Develop New PVT Correlations for Saudi Crude Oils, in: Abu Dhabi Int. Pet. Exhib. Conf., Society of Petroleum Engineers, 2002.

[47] V. Gaganis, N. Varotsis, Machine Learning Methods to Speed up Compositional Reservoir Simulation, in: 74th EAGE Conf. Exhib. SPE EUROPEC 2012, no. June 2012, Society of Petroleum Engineers, Copenhagen, Denmark, 2012, pp. 4–7.

[48] J. O. Valderrama, A. Reategui, R. E. Rojas, Density of Ionic Liquids Using Group Contribution and Artificial Neural Networks, Ind. Eng. Chem. Res. 48 (6) (2009) 3254–3259.

[49] J. O. Valderrama, C. A. Fau, V. J. Vicencio, Artificial Neural Networks and the Melting Temperature of Ionic Liquids, Ind. Eng. Chem. Res. 53 (2014) 10504–10511.

[50] M. Hosseinzadeh, A. Hemmati-Sarapardeh, Toward a predictive model for estimating viscosity of ternary mixtures containing ionic liquids, J. Mol. Liq. 200 (PB) (2014) 340–348.

[51] P. Díaz-Rodríguez, J. C. Cancilla, G. Matute, J. S. Torrecilla, Viscosity estimation of binary mixtures of ionic liquids through a multi-layer perceptron model, J. Ind. Eng. Chem. 21 (2015) 1350–1353.

[52] J. Hekayati, M. R. Rahimpour, Estimation of the saturation pressure of pure ionic liquids using MLP artificial neural networks and the revised isofugacity criterion, J. Mol. Liq. 230 (2017) 85–95.

[53] A. A. Rohani, G. Pazuki, H. A. Najafabadi, S. Seyfi, M. Vossoughi, Comparison between the artificial neural network system and SAFT equation in obtaining vapor pressure and liquid density of pure alcohols, Expert Syst. Appl. 38 (3) (2011) 1738–1747.

[54] F. Gharagheizi, A. Eslamimanesh, B. Tirandazi, A. H. Mohammadi, D. Richon, Handling a very large data set for determination of surface

tension of chemical compounds using Quantitative Structure-Property Relationship strategy, Chem. Eng. Sci. 66 (21) (2011) 4991–5023.

[55] S. S. Sablani, O.-D. Baik, M. Marcotte, Neural networks for predicting thermal conductivity of bakery products, J. Food. Eng. 52 (3) (2002) 299–304.

[56] M. S. Rahman, M. M. Rashid, M. A. Hussain, Thermal conductivity prediction of foods by Neural Network and Fuzzy (ANFIS) modeling techniques, Food Bioprod. Process. 90 (2) (2012) 333–340.

[57] J. E. Schmitz, R. J. Zemp, M. J. Mendes, Artificial neural networks for the solution of the phase stability problem, Fluid Phase Equilib. 245 (1) (2006) 83–87.

[58] S. Mohanty, Estimation of vapour liquid equilibria of binary systems, carbon dioxide-ethyl caproate, ethyl caprylate and ethyl caprate using artificial neural networks, Fluid Phase Equilib. 235 (1) (2005) 92–98.

[59] B. Vaferi, Y. Rahnama, P. Darvishi, A. Toorani, M. Lashkarbolooki, Phase equilibria modeling of binary systems containing ethanol using optimal feedforward neural network, J. Supercrit. Fluids 84 (2013) 80–88.

[60] A. Farzi, A. T. Nejad, Prediction of phase equilibria in binary systems containing acetone using artificial neural network, Int. J. Sci. Eng. Res. 6 (9) (2015) 5–10.

[61] M. Moghadam, S. Asgharzadeh, On the application of artificial neural network for modeling liquid-liquid equilibrium, J. Mol. Liq. 220 (2016) 339–345.

[62] K. T. Butler, D. W. Davies, H. Cartwright, O. Isayev, A. Walsh, Machine learning for molecular and materials science, Nature 559 (7715) (2018) 547–555.

[63] COCO Help: TEA, `https://www.cocosimulator.org/index_help.php?page=TEA/tea.htm`, 2018 (accessed 4 October 2018).

[64] COCO - the CAPE-OPEN to CAPE-OPEN simulator, `https://www.cocosimulator.org/index.html`, 2018 (accessed 6 September 2018).

[65] G. Soave, Equilibrium constants from a modified Redlich-Kwong equation of state, Chem. Eng. Sci. 27 (6) (1972) 1197–1203.

[66] D. Y. Peng, D. B. Robinson, A New Two-Constant Equation of State, Ind. Eng. Chem. Fundam. 15 (1) (1976) 59–64.

[67] J. Gross, G. Sadowski, Perturbed-chain SAFT: An equation of state based on a perturbation theory for chain molecules, Ind. Eng. Chem. Res. 40 (4) (2001) 1244–1260.

[68] G. M. Wilson, Vapor-Liquid Equilibrium. XI. A New Expression for the Excess Free Energy of Mixing, J. Am. Chem. Soc. 86 (2) (1964) 127–130.

[69] H. Renon, J. M. Prausnitz, Local compositions in thermodynamic excess functions for liquid mixtures, AIChE J. 14 (1) (1968) 135–144.

[70] D. S. Abrams, J. M. Prausnitz, Statistical thermodynamics of liquid mixtures: A new expression for the excess Gibbs energy of partly or completely miscible systems, AIChE J. 21 (1) (1975) 116–128.

[71] M. L. Michelsen, Phase Equilibrium Calculations. What is Easy and What is Difficult?, Comput. Chem. Eng. 17 (5/6) (1993) 431–439.

[72] R. Rojas, Neural Networks: A Systematic Introduction, 1st Edition, Springer-Verlag, Berlin; Heidelberg, 1996.

[73] D.-A. Clevert, T. Unterthiner, S. Hochreiter, Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs), in: ICLR 2016 Conf., 2015, pp. 1–14.

[74] B. Xu, N. Wang, T. Chen, M. Li, Empirical Evaluation of Rectified Activations in Convolutional Network (2015) 1–5.

[75] G. Cybenko, Approximation by superpositions of a sigmoidal function, Math. Control Signals Syst. 2 (4) (1989) 303–314.

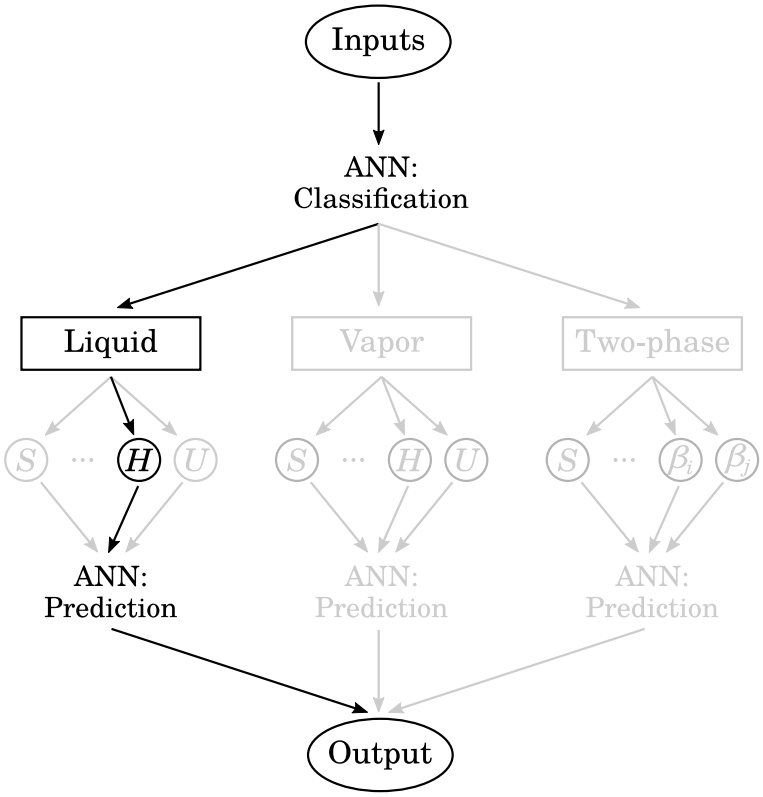
[76] K. Hornik, Approximation capabilities of multilayer feedforward networks, Neural Networks 4 (2) (1991) 251–257.

[77] S. Sonoda, N. Murata, Neural network with unbounded activation functions is universal approximator, Appl. Comput. Harmon. Anal. 43 (2) (2017) 233–268.

[78] J. F. Boston, H. I. Britt, A radically different formulation and solution of the single-stage flash problem, Comput. Chem. Eng. 2 (2-3) (1978) 109–122.

[79] Keras Documentation, `https://keras.io/`, 2018 (accessed 09 September 2018).

[80] T. Dozat, Incorporating Nesterov Momentum into Adam, Tech. Rep. 1 (2016).



Graphical abstract