University of Cape Town

Department of Chemistry

Prediction of Isobaric Heat Capacities of Room Temperature Ionic Liquids

Tayla Lee Wilson

Supervisor: Dr. Gerhard A. Venter

Submitted in fulfilment of the requirements for the degree of MSc Chemistry

October, 2020

Declaration:

1. I know that plagiarism is wrong. Plagiarism is to use another's work and to pretend that it is one's own.
2. I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as his or her own work.
3. This project is my own work.
4. I have included internet article, book, or other material references used for this project.

Signed:

Signed by candidate

Date: Friday 16th October, 2020

Abstract

Ionic liquids (ILs) have many potential applications that require knowledge of a variety of physical and thermodynamic properties. While these properties can often be determined experimentally, this is impossible for novel, yet to be synthesised ILs; thus, property prediction from first principles is essential to unlock new developments in the rational design of ILs. The isobaric heat capacity (C_P) is an important thermodynamic property that quantifies the amount of heat needed to increase the temperature of a material and is thus of great importance in engineering applications involving the design of heat-transfer systems. From a theoretical viewpoint, the heat capacity is a fundamental quantity that expresses the temperature dependence of enthalpy and entropy. Several models for the prediction of C_P have been developed to date; however, these are often trained on limited data sets and published model performance is largely dependent on the judicious choice of the testing data. Moreover, popular techniques such as group contribution methods (GCMs) cannot always be applied to structurally novel ILs and quantitative structure property relationships (QSPRs) are highly dependent on the diversity of training data.

In this work, predictive models for C_P have been developed using linear and nonlinear machine-learning methods. A large data set of 2463 temperature-dependent C_P values, spanning 208 ILs, was obtained from the ILThermo database. Molecular volumes, features based on the electrostatic potential (ESP) and other molecular descriptors were calculated for each cation and anion in the data set. Following this, several multiple linear regression models were developed, for which Lasso regression was used reduce the number of features, where necessary. The models were developed using a methodology that attempts to reduce the dependency of the results on the identity of the specific species in the training set. The complexity of these models was gradually increased from a simple volume-based model (inspired by the success of the Volume Based Thermodynamics (VBT) approach of Glasser and Jenkins [L. Glasser and H. D. B. Jenkins, *Chem. Soc. Rev.*, 2005, **34**, 866], which was applied to ionic liquids and augmented by Krossing and co-workers [W. Beichel

et al., *J. Mol. Liq.*, 2014, **192**, 3]), to the addition of electrostatic potential surface areas and finally including the General Interaction Properties Functions (GIPFs) of Murray and Politzer [J. S. Murray et al., *J. Mol. Struct. (THEOCHEM)*, 1994, **307**, 55], which are statistically well-defined quantities derived from ESP data, and a Feed Forward Neural Network (FFNN) was developed using the most effective of the aforementioned feature sets. In addition to reporting test-set errors, an external data set was carefully compiled, containing ILs with components (either the cation or anion) not present in the training data, and structurally distinct. This was done to assess the general applicability and flexibility of the final models, and to allow for a fair comparison of model performance. Of the linear models developed, that using interacting features consisting of molecular volumes and GIPFs produced the lowest errors; this is likely due to the ability of the interaction features to describe intermolecular interactions between cations and anions. Consequently, molecular volumes and GIPFs features were also used to develop a nonlinear FFNN. Finally, the linear interacting GIPFs model and FFNN also produced the lowest errors of $3.2 \pm 1.5\,\%$ and $3.8 \pm 2.4\,\%$, respectively, when applied to the external data set.

Acknowledgements

I would first like to thank my supervisor, Dr. Gerhard Venter, for the guidance, mentorship and many invaluable discussions over the past two years.

I would also like to thank the Scientific Computing Research Unit for laboratory use and the group members for the many necessary coffee breaks!

Acknowledgement is given to the Centre for High-Performance Computing (CHPC), South Africa, for providing computational resources to this research project.

I would like to express my deepest gratitude to my parents, Glisson, Lynne and Shelagh, for all the love and support throughout my entire degree.

To my brother Bradlee, thank you for all the help over the years, I would not have made it this far without you!

Finally, thank you, Nicole, for the motivation while writing up and thank you, Nikola, for your continuous encouragement and support.

Contents

1 Introduction **1**

1.1 Introduction to Ionic Liquids 1

1.1.1 Physical Properties of Ionic Liquids 4

1.1.2 Applications of Ionic Liquids 14

1.2 Introduction to Heat Capacities 15

1.3 Predicting Heat Capacities 19

1.3.1 Group Contribution Methods 20

1.3.2 Quantitative Structure-Property Relationship Methods 27

1.3.3 Volume-Based Thermodynamics 30

1.3.4 Nonlinear Methods 34

1.4 Aims and Objectives 36

2 Theoretical Background **38**

2.1 Heat Capacity 38

2.1.1 Isochoric Heat Capacity 39

2.1.2 Isobaric Heat Capacity 39

2.1.3 Relating Isobaric and Isochoric Heat Capacities 40

2.1.4 Obtaining the Isobaric Heat Capacity 42

2.1.5 Isobaric Heat Capacity and Temperature Dependence of State Functions 44

2.2 Machine Learning Theory 45
2.2.1 Linear Regression 46
2.2.2 Regularised Linear Regression 48
2.2.3 Artificial Neural Networks 50
2.3 QM Theory 56
2.3.1 Hartree-Fock Theory 56
2.3.2 Density Functional Theory 59

3 Methods and Model Development 62
3.1 Data Collection and Preparation 63
3.2 Descriptors 65
3.2.1 Structure Generation 66
3.2.2 Molecular Volumes 67
3.2.3 Temperature 70
3.2.4 Electrostatic Potential-based features 71
3.2.5 Molecular Descriptors 73
3.2.6 Feature Scaling 73
3.3 Model Assessment 74
3.4 Linear Regression Model Development 75
3.4.1 Train:Test Split 75
3.4.2 Feature reduction 76
3.5 Feed Forward Neural Network Development 77
3.5.1 Train:Test split 77
3.5.2 Hyperparameter optimisation 79
3.6 External Data Set 81

4 Predicting Heat Capacities 87
4.1 Linear Regression 87

4.1.1 Baseline Model 88
4.1.2 Electrostatic Potential Surface Areas 92
4.1.3 General Interaction Properties Functions 95
4.1.4 QSPR Model 101
4.2 Feed-Forward Neural Network 108
4.2.1 Architecture 109
4.2.2 FFNN Optimisation 109
4.3 Summary of Results 110
4.4 Model Comparison 111

5 Conclusion **115**

A Linear Regression **130**

B Lasso Regularisation **133**

C Feed Forward Neural Network **137**

Chapter 1

Introduction

This chapter contains three sections: First, an introduction to ionic liquids (ILs) is given, including a brief history and discussion of a suitable definition, as well as a description of selected physical properties and structural characteristics, followed by a brief overview of some applications of these fluids. The second section then focuses specifically on the property of interest of this work, the heat capacity. The next section presents a review of relevant literature in which heat capacities of ILs have been predicted using various methods, including both linear and nonlinear regression. Lastly, the aims and objectives of the project are given in the final section.

1.1 Introduction to Ionic Liquids

Paul Walden is typically credited with synthesising the "first" IL in 1914.[1] He investigated the relationship between molecular size and conductivity and consequently synthesised the protic IL ethylammonium nitrate, of which the melting point was determined to be between 13–14 °C. This brought to light the effect of a fairly large cation on lowering the melting temperature of salts, a point that will be expanded on later. Low-melting salts held great potential, which went largely untapped until 1951. Hurley and Wier[2] recognised the potential of these fluids and used mixtures of ILs (consisting of pyridinium-based cations

and halide anions) and aluminium chloride as electrolyte solutions for the electroplating of metals. From this point onwards, a notable amount of work was carried out on systems containing mixtures of ILs and aluminium chloride, proving to have several potential uses as solvents in electrochemistry, synthesis, and spectroscopy.[3–6] A concern with aluminium chloride systems is the extreme sensitivity to water, requiring careful consideration when handling; however, in 1983, Cooper and Angell[7] made the first report of a water-stable IL, methoxyethyl dimethyl ethylammonium tetrafluoroborate with a melting point of $13\,^{\circ}\mathrm{C}$. This was followed by the report of methoxymethyl dimethyl ethylammonium tetrafluoroborate, with a melting point of $-16\,^{\circ}\mathrm{C}$, in 1986 by the same group.[8] Although these reports were recognised as important developments in the field of ILs, there was no follow up until six years later. In 1992, Wilkes and Zaworotko[9] synthesised salts comprised of the 1-ethyl-3-methylimidazolium cation and anions including tetrafluoroborate, nitrate, and acetate. It was noted that these ILs appear to be air and water stable, making them more favourable than the previously mentioned tetrachloroaluminate-based ILs; however, it is now known that fluorinated anions such as tetrafluoroborate hydrolyse forming toxic hydrofluoric acid.[10] This lead to a peaked interest in these systems in recent years, and ILs have emerged as promising materials with many potential applications in synthesis and catalysis,[11–15] driving research into these compounds.

At this point it is necessary to clarify what these species are. Ionic liquids have been broadly defined as molten salts, comprised entirely of ions, that are liquid at temperatures lower than $100\,^{\circ}\mathrm{C}$.[16] It is common to further classify such species that are liquid at low temperatures as ambient-temperature or room-temperature ILs (RTILs).[17] The inclusion of a melting temperature in the definition allows for a clear distinction between "ionic liquids" and high-melting salts such as NaCl, which has a melting point of $\approx 800\,^{\circ}\mathrm{C}$. The low-melting nature is advantageous from a practical sense in that they are easier to handle than higher melting salts.[18] However, this definition of an RTIL is an arbitrary and limiting classification, as this does not include species with melting points between $100\,^{\circ}\mathrm{C}$

and that of crystalline salts, which could still have useful applications. Nonetheless, recent literature often refers to RTILs and it is therefore necessary to define the term (see MacFarlane et al.[19] for a discussion of various definitions that can be used for ILs). Frequently used RTILs consist of a large, usually nitrogen-containing and asymmetric, organic cation, and an organic or inorganic anion,[20] and it is this requirement that best defines the term "ionic liquid" as used in this work. The names of ILs can become complex and long, and abbreviations are often used. Abbreviations of ions are generally consistent, but can differ across references, such as seen for 1-ethyl-3-methylimidazolium. This is a widely used cation, for which abbreviations include $[C_2C_1im]^+$, $[C_2MIM]^+$ or $[emim]^+$, as used respectively by Welton, Preiss et al. and Valderrama et al., amongst others.[11,21–23] For consistency, ions are abbreviated here in the same general form as suggested by Welton and a list of preferred abbreviations has been given by Kar et al.[24] It should be noted that the charge is only indicated when the ions are unpaired, but when an ion pair is abbreviated no charge is indicated; for example, 1-ethyl-3-methylimidazolium acetate is abbreviated as $[C_2C_1im][OAc]$. The structures of some common cations and anions used in ILs, as well as the abbreviations used, are shown in Figures 1.1 and 1.2, respectively.

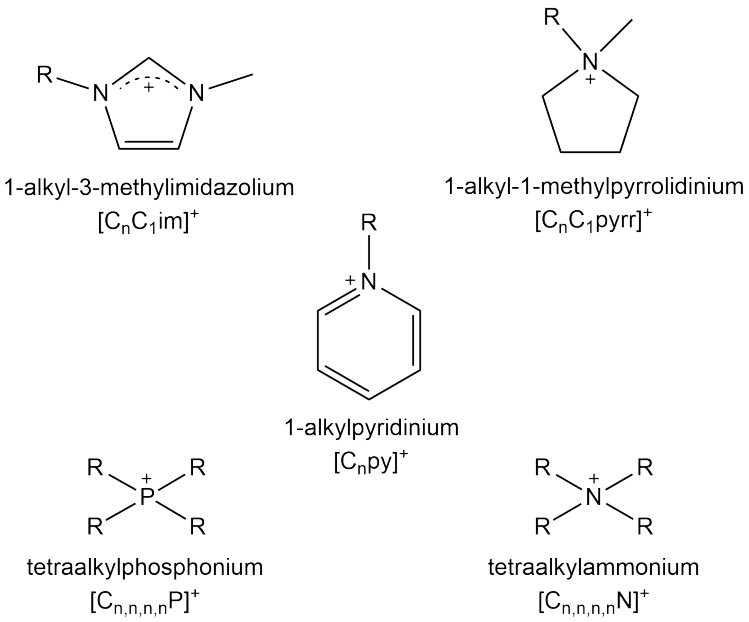


Fig. 1.1. Examples of typical core cation structures of RTILs.

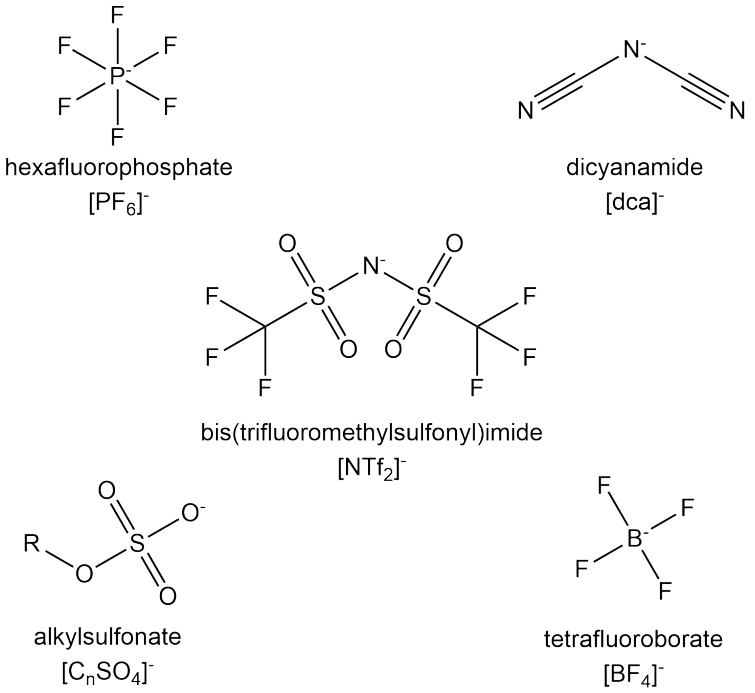


Fig. 1.2. Examples of typical anion structures of RTILs.

RTILs have many favourable physical properties, detailed in the section below, which are in turn due to the structural characteristics of ILs, such as being comprised entirely of ions and the nature of these ions. The focal point of research into RTILs is the significant variance in physical properties based on the identity of the cation and anion. In addition, cations and anions can be functionalised to impart specific chemical properties; for example, the inclusion of Brønstead acidic groups.[25] This introduces the possibility of designing RTILs with the required properties that are "tailor-made" for a particular application, resulting in so-called task-specific ILs (TSILs).[20] Properties and characteristics of interest, as well as some potential applications of RTILs, will be discussed in the next subsection.

1.1.1 Physical Properties of Ionic Liquids

Appealing properties of RTILs include the following: low melting points, high viscosities, relatively low vapor pressures, thermal stability, and broad/extended electrochemical stability. To understand these properties first requires the determination of these properties.

The melting point is possibly the most essential to quantify to ensure that a species is in fact a liquid at room temperature, as this is the fundamental quality of an RTIL. Viscosity is also a useful property to obtain and understand, as the lowering of RTIL viscosities can enhance the transport properties, namely ion and thermal conductivity, of these species. The low vapour pressures of RTILs is an attractive quality from a safety and "green chemistry" point of view—low volatility results in low vapour emissions of the species into the atmosphere.[26] Finally, thermal and electrochemical stability are key properties as they allow for many electrochemical and thermal applications. It is therefore necessary to be able to accurately quantify these properties for the rational design of novel TSILs, which requires an understanding of the factors influencing these properties.

1.1.1.1 Melting Point

The melting point can be understood in terms of the liquid range, which is the temperature region between crystallization and thermal decomposition of an IL; in other words, it is the temperature range over which a salt remains a liquid. The melting point is the lower boundary of the liquid range and is the temperature at which the long-range order of the solid phase is broken. Melting points are typically measured using differential scanning calorimetry (DSC), but accurate measurements of melting points of RTILs are difficult, as impurities can affect the melting temperature. It is not only the presence of impurities make these measurements challenging—the presence of polymorphs can make the characterization of the melting point complicated.[27] Therefore, predictive methods are necessary to obtain melting points, and have be used for the prediction for RTILs and previous prediction methods have been reviewed by Rooney et al.[28]

RTILs have low melting points compared to that of inorganic salts and this can be attributed to the unique structural features of these species. While the melting point depends on properties of both the solid and liquid phase, it is mostly a reflection of the interactions present in the crystal phase, which is in turn a reflection of the free energy, G. If there are

two possible states, the one with the lower free energy is favoured, as it is more stable. At temperatures below the melting point, T_{m}, the free energy of the crystalline phase, G_{c}, is lower than the free energy of the liquid phase, G_{l}, and at temperatures above T_{m} the opposite is true. As indicated in Figure 1.3, T_{m} is the point at which the two phases are in equilibrium as $G_{\mathrm{c}} = G_{\mathrm{l}}$, and $\Delta G = 0$. Using the Gibbs-Helmholtz equation,

$$\Delta G = \Delta H - T\Delta S, \tag{1.1}$$

T_{m} can be expressed as T_{fus}, such that

$$T_{\mathrm{fus}} = \frac{\Delta_{\mathrm{fus}} H}{\Delta_{\mathrm{fus}} S}. \tag{1.2}$$

This shows that T_{fus} can be depressed by decreasing the enthalpy of fusion, $\Delta_{\mathrm{fus}} H$, and/or

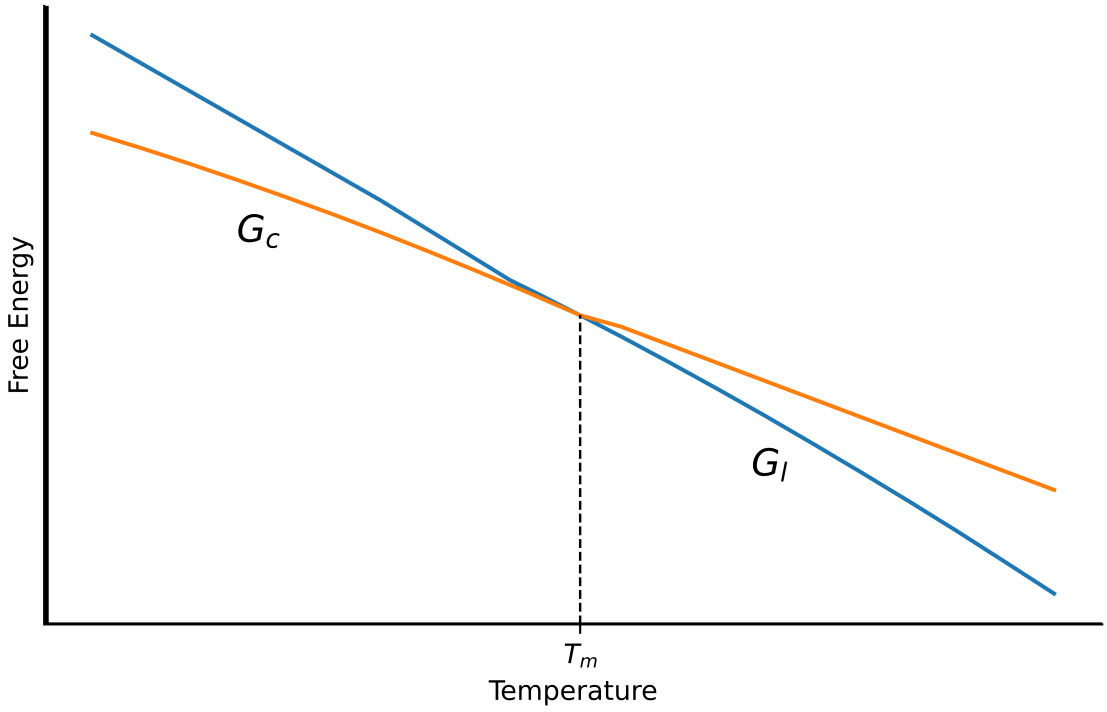


Fig. 1.3. Typical free energy curve indicating the free energy, G, for the crystalline phase (G_{c}) indicated in orange and the liquid phase (G_{l}) indicated in blue. T_{m} indicates the melting point.

increasing the entropy of fusion, $\Delta_{\text{fus}}S$. The former can be achieved by weakening intermolecular interaction in the solid phase and thus destabilizing the crystal lattice. $\Delta_{\text{fus}}S$ is a reflection of the disorder of the crystalline state and is increased with configurational degrees of freedom: Entropy, S, can be expressed as

$$S = k_B \ln (W), \tag{1.3}$$

where W is the number of microstates, which is the number of ways the ions can be arranged to give the same configuration. As the number of uniquely positioned atoms increases, W will decrease leading to a lower solid-phase entropy; however, the liquid phase entropy of a species will remain the same, regardless of how it crystallises, meaning that $\Delta_{\text{fus}}S$ of a species is only affected by the crystal-phase entropy. Therefore, when the crystal state consists of ions in positions that vary greatly, the ions have a lower entropy in the crystal state and a larger increase upon melting.[19] Consequently, more configurational freedom results in a larger $\Delta_{\text{fus}}S$.

Based on this, there are three factors that influence the free energy, which need to be considered to understand melting points: intermolecular forces, molecular symmetry and internal degrees of freedom. The structural features affecting these factors will be discussed.

Intermolecular Interactions. To rationalise the strength of the intermolecular interactions present among RTILs, electrostatic interactions need to be considered. ILs experience many attractive forces including Van der Waals interactions, hydrogen bonding, and π-π interactions; however, it is the electrostatic forces present which make the largest contribution to the overall interactions experienced by these species. The electrostatic force, F, can be used to quantify the strength of the interaction between two charged species, and

is determined using Coulomb's Law,

$$F = \frac{kq_1q_2}{r^2}, \tag{1.4}$$

where k is Coulomb's constant, q_1 and q_2 are the species' charges, and r is the distance between the centers of charge. The attractive force between ions decreases as the distance between the centres of charge increases. Therefore, when larger molecular ions are used, the minimum energy will occur at a larger distance between the charged regions of the ions; consequently, weaker electrostatic interactions will result, favourable for ILs typically consisting of larger cations such as those depicted in Figure 1.1. The strength of the electrostatic interactions is also influenced by charge delocalisation, driven by factors including the inductive effect and resonance stabilisation. The inductive effect is due to the presence of highly electronegative atoms in a molecule, such as hexafluorphosphate where the electronegative fluorine atoms draw charge away from the phosphate centre, delocalising the net charge. Resonance plays a role as when an ion is resonance stabilised, as seen with bistriflimide, it leads to the spread of the net charge over many atoms. The effect of charge delocalisation can be illustrated with the following: 1-ethyl-3-methylimidazolium chloride ([C_2C_1im][Cl]) and 1-ethyl-3-methylimidazolium bromide ([C_2C_1im][Br]) consist of anions with very localised charge and have melting points at temperatures much higher than 1-ethyl-3-methylimidazolium bistriflimide ([C_2C_1im][NTf_2]) and 1-ethyl-3-methylimidazolium hexafluorophosphate ([C_2C_1im][PF_6]) (See Table 1.1). As lower melting points are favourable, typical anions used generally experience charge delocalisation, as those shown in Figure 1.2.

In addition, electrostatic interactions can also be weakened through shielding due to the presence of nonpolar groups, as this decreases contact between the charged regions. 1-ethyl-3-methylimidaozlium hexafluorophosphate ([C_2C_1im][PF_6]), 1-propyl-3-methylimidazolium hexafluorophosphate ([C_3C_1im][PF_6]), and 1-butyl-3-

methylimidazolium hexafluorphosphate ([C_4C_1im][PF_6]) illustrate the effect of shielding. These three species differ only by length of alkyl chain and, as seen in Table 1.1, as the alkyl chain gets longer the melting point decreases due to the increased shielding, and consequently, IL cations often have bulky alkyl chains. Although bulkier alkyl groups decrease melting points, these tend to increase when the chain is more than $12\,\text{Å}$ in length from the charged region (approximately seven carbons in length). This can be understood by considering the structural regions as described by López-Martin et al.[29] illustrated in Figure 1.4. When an alkyl chain is of a length that falls within the symmetry-breaking region, an increase in length will lower the melting point of a species; however, when the alkyl chain falls in the hydrophobic region, increasing the chain length will result in a higher melting point as the Van der Waals interactions become more significant, increasing the strength of the intermolecular interactions.

The length of the alkyl chain also influences the extent to which polar and nonpolar structural domains aggregate, leading to nanostructuring in the liquid, which has been confirmed both experimentally[30] and using molecular dynamics simulations.[31] Canongia Lopes and Pádua[31] showed that in imidazolium-based ionic liquids, the polar head groups of the cation aggregate to form a three-dimensional network, whereas the nonpolar domains are arranged in dispersed clusters. However, as the alkyl chain length increases to beyond four carbon atoms, the nonpolar domains become continuous. The authors attributed the behaviour of properties such as viscosity and conductivity, which shows a break in trend between ILs with short (C_1 and C_2) and long (C_4 and longer) chains,[14] to the existence of these microdomains. The characterisation of liquid phase structure and nanostructure in ILs and the influence of structuring on properties is an active area of research, which has recently been reviewed by Hayes et al.[32]

Ion Asymmetry. Another factor influencing the melting point is the crystal packing of the ions. Symmetrical ions will pack more efficiently resulting in a more stable crystal lat-

tice, whereas asymmetrical ions will result in a less ordered crystal structure, decreasing the packing efficiency, and consequently the melting point. Considering 1,2,3,4,5-pentamethylimidazolium bis(trifluoromethylsulfonyl)imide ($[C_1C_1C_1C_1C_1im][NTf_2]$), 1,2-dimethyl-3-propylimidazolium bis(trifluoromethylsulfonyl)imide ($[C_3C_1C_1im][NTf_2]$), and 1-butyl-3-methylimidazolium bis(trifluoromethylsulfonyl)imide ($[C_4C_1im][NTf_2]$), the effect of ion symmetry can be illustrated. These liquids consist of the same anion, and cations of the same empirical formula that differ in connectivity of the alkyl groups to the imidazolium core. It can be seen in Table 1.1 that the $[C_1C_1C_1C_1C_1im]^+$ ion is the most symmetrical, resulting in a higher melting point than the less symmetrical $[C_3C_1C_1im]^+$ and $[C_4C_1im]^+$ ions.

Degrees of Freedom. The melting point is also affected by the entropy of the crystalline state, which is dependent on the degrees of freedom of the molecule. If an ion has multiple low-energy conformations, it is possible that it can crystallize in a less favourable conformation, resulting in a less stable crystal lattice. If more than one conformation is present in the solid state, this will impact the packing efficiency, contributing to lower melting points.[19]

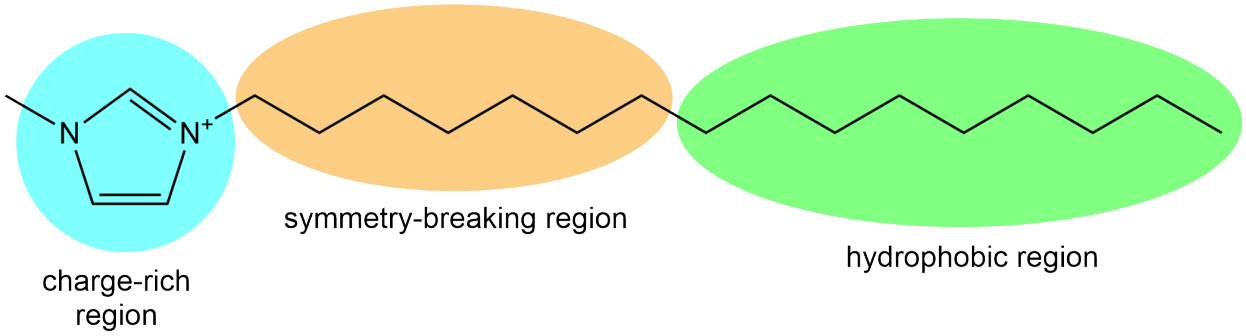
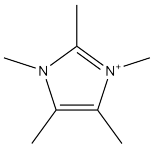
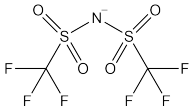
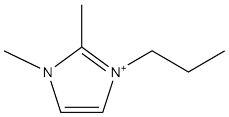
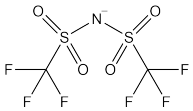
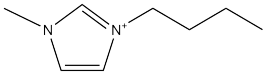
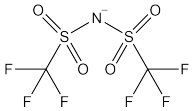
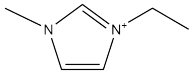
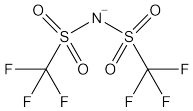
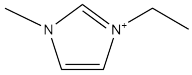
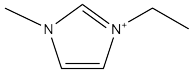
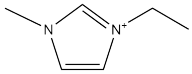
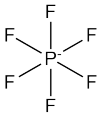
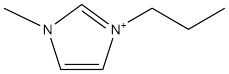
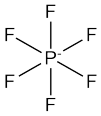
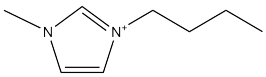
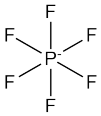


Fig. 1.4. Illustration of the structural regions defined by López-Martin et al.[29] for the 1-hexadecyl-3-methylimidazolium cation.

Table 1.1. Melting points (T_m) of select RTILs.[a]

RTIL	Cation structure	Anion structure	T_m
$[C_1C_1C_1C_1C_1im][NTf_2]$			391.1
$[C_3C_1C_1im][NTf_2]$			288.1
$[C_4C_1im][NTf_2]$			268.5
$[C_2C_1im][NTf_2]$			274.1
$[C_2C_1im][Br]$		Br^-	352.1
$[C_2C_1im][Cl]$		Cl^-	362.1
$[C_2C_1im][PF_6]$			333.0
$[C_3C_1im][PF_6]$			311.6
$[C_4C_1im][PF_6]$			282.2

[a] Melting points in K. Values are obtained from ILThermo.

1.1.1.2 Viscosity

Viscosity is a rheological property which expresses the resistance of a fluid against deformation. ILs have relatively high viscosities compared to that of typical organic solvents, as shown in Table 1.2. For example, common solvents such as pyridine have viscosities below $1\,\mathrm{cP}$, having consistencies similar to that of water. ILs are far more viscous, with viscosities more than ten times that of water, as seen in Table 1.2. Viscosities vary among ILs, and many typically have consistencies between that of sunflower oil and glycerol; however, species with higher viscosities are also seen. The reason for the high viscosities is largely driven by the size of the ions and the strong interionic interactions—liquids consisting of larger ions tend to be more viscous, a trend seen among the ILs in Table 1.2. Reducing viscosities is an important research area in this field, as lower viscosities would improve aspects such as ion transport, favourable in applications such as lithium ion batteries.[19] Viscosities are usually measured with falling ball, capillary or rotational viscometers, each of which is explained by Wasserscheid and Welton.[33] While each of these methods can provide reliable measurements, the presence of impurities can lead to inaccuracies, and many correlation methods have been used for the prediction of IL viscosities, reviewed by Wang et al.[34]

1.1.1.3 Vapour Pressure

One of the defining characteristics of ILs is very low vapour pressures. Solvents with low vapour pressures are sought after, as the release of these compounds into the environment is less than that of more volatile, traditional organic solvents.[38] Whereas it was initially assumed that ILs decompose at high temperatures and could not be distilled, it was shown by Earle et al.[39] that it is indeed possible. The ability of ILs to vaporise allows for the measurement of the heat of vaporisation, $\Delta_{\mathrm{vap}}H$, a fundamental thermodynamic property which can be used to understand liquid-phase interactions.

Table 1.2. Viscosities (η) for selected liquids.[a]

Liquid	η (cP)
Acetone	0.318
Toluene	0.59
Benzene	0.652
Pyridine	0.974
Water	1
$[C_3C_1pyrr][N(CN)_2]$	45
Sunflower oil	48.8[35]
$[C_4C_1pyrr][N(CN)_2]$	50
$[C_4C_1py][NTf_2]$	63
$N_{1113}][NTf_2]$	72
$[C_6C_1py][NTf_2]$	85
$[C_8C_1py][NTf_2]$	112
Maple Syrup	138[36]
$[C_4C_1im][BF_4]$	154
$[N_{2226}][NTf_2]$	167
$[N_{2228}][NTf_2]$	202
$[C_6C_1im][BF_4]$	314
$[C_8C_1im][BF_4]$	439
$[C_6C_1im][PF_6]$	690
$[C_8C_1im][PF_6]$	866
Glycerol	965.8[37]

[a] Viscosities in centipoise (cP) at 298.15 K. Viscosities taken from Reference 33 unless otherwise indicated.

1.1.1.4 Electrochemical Properties

Electrochemical properties such as electrochemical stability and high ionic conductivity are notable characteristics of ILs that have allowed for many electrolytic applications. Electrochemical stability is determined by the electrochemical window (ECW)—the potential range over which an electrolyte is stable to reduction and oxidation. As cations would be reduced and anions oxidised, the ECW of an IL is the difference between the reduction potential of the cation and the oxidation potential of the anion. The ECW of ILs is typically larger than that of traditional aqueous electrolytes as the ions are more stable, having higher oxidative and reductive limits.[33] A high ECW therefore indicates greater electrochemical stability, contributing to the success of ILs in applications, such as battery

electrolytes and electrodeposition of metals, as reviewed by Armand et al.[40]

1.1.1.5 Thermal Properties

Thermal properties are related to conduction of heat and those of interest for ILs include thermal conductivity and heat capacity. The thermal conductivity of an RTIL is the rate of heat flow through the liquid as a result of a temperature gradient,[19] for which there are limited experimental values as highlighted by Rooney et al.;[28] therefore limited work has been done on the prediction of thermal conductivities. The heat capacity of an RTIL is the amount of heat that is required to result in a unit temperature change, related to thermodynamic properties such as enthalpy and entropy; consequently it can be used to explain and model many other properties of RTILs, making it an important property to measure and predict. Measurement can be done using calorimetry methods, and predictive models have been developed, and are reviewed here in detail (see Section 1.3).

1.1.2 Applications of Ionic Liquids

There are many potential applications of these liquids for which a notable few will be addressed here. RTILs have been studied as alternatives to traditional organic solvents and due to the low volatilities have been considered as potential "green solvents". The varying nature of ILs, resulting from many structural variations, has lead to several solvation applications. Due to the ionic nature, not only do the ions exert strong electrostatic interactions on each other, but also on the molecules of the solute. Certain functional groups present on the cation and anion assist with the dissolution of polar organic compounds, and long alkyl chains commonly found on cations improve the dissolution of nonpolar organic compounds.[19]

A high impact application of ILs receiving much attention is that of biomass dissolution/swelling. Biomass, such as cellulose, chitin and chitosan, is a renewable resource with many applications including biofuels and biomaterials and ILs are proposed as sol-

vents for the extraction and purification of active compounds from biomass.[41] ILs are also advantageous in biphasic catalysis, as they can be entrapped allowing for extraction of organic products and the reusing of the IL solvent.[42] Gasses such as carbon dioxide, nitrogen and methane are all soluble in ILs, and due to the ability of ILs to capture CO_2, focus has been placed on using these species as this is an important process applicable to the reduction of greenhouse gasses.[43]

Another active area of research in the field of ILs is for use in energy storage applications. Due to the high ionic conductivity and electrochemical stability, ILs have been proposed as electrolytes in high-energy electrochemical devices such as lithium, sodium, magnesium, and metal-air batteries.[14,44,45] RTILs have been used in fuel cells for proton transport, in place of hydronium or hydroxide ions, allowing for a greater selection of electrolytes, not limited to typical aqueous electrolytes. Use of RTILs as electrolytes is advantageous owing to the large ECW allowing for conductivity under many temperature conditions. The high ECW high thermal stability of RTILs are favourable properties for thermoelectrical cells, which convert thermal energy is to electrical energy, and using RTILs in place of the typically used water-based electrolytes allows for access to a wider temperature range of heat sources.[45]

ILs are good solvents for material synthesis, largely nanomaterial synthesis. The high ECW and ionic conductivity allows for the preparation of nanostructures through electrodeposition using RTIL electrolytes. RTILs can stabilise nanoparticles during synthesis due to the alkyl chains of the cations and certain functional groups present in both cations and anions.[46]

1.2 Introduction to Heat Capacities

This section provides an overview of the heat capacity, with a focus on its practical definition and application in thermodynamics, as well as a brief discussion of how experimental

measurement can be done, and the quantities obtained in doing so. The theoretical development of some of the essential relationships stated here are given in Chapter 2.

The heat capacity, C, is defined as the heat flow required to change the temperature of a given amount of material by one temperature unit,

$$C = \frac{đq}{dT} \tag{1.5}$$

In this expression, C is an extensive thermodynamic quantity; however, it is typically reported as an intensive quantity, either as the molar heat capacity (C_m), in units of $\mathrm{J/(mol\,K)}$, or specific heat capacity (c) in units of $\mathrm{J/(g\,K)}$,

$$C_\mathrm{m} = \frac{C}{n}, \tag{1.6a}$$

$$c = \frac{C}{m}, \tag{1.6b}$$

where n and m are the amount of moles and mass, respectively, of the material. *Note that the subscript "m" is often omitted in literature when molar heat capacities are referred to, which is the convention applied here.*[47] Furthermore, since the amount of heat required is path-dependent, two heat capacities are commonly encountered when dealing with homogeneous fluids: the isobaric heat capacity (C_P), which measures heat flow along a constant pressure path (q_P) and the isochoric heat capacity (C_V), which measures heat flow along a constant volume path (q_V),

$$C_P = \frac{đq_P}{dT} \tag{1.7a}$$

$$C_V = \frac{đq_V}{dT}. \tag{1.7b}$$

The heat capacity is of fundamental importance in chemical thermodynamics as it can be used to describe the temperature dependence of all of the key thermodynamic state

functions,[48] as shown in

$$C_P = \left(\frac{\partial H}{\partial T}\right)_P = T\left(\frac{\partial S}{\partial T}\right)_P = -T\left(\frac{\partial^2 G}{\partial T^2}\right)_P \tag{1.8}$$

and

$$C_V = \left(\frac{\partial U}{\partial T}\right)_V = T\left(\frac{\partial S}{\partial T}\right)_V = -T\left(\frac{\partial^2 A}{\partial T^2}\right)_V, \tag{1.9}$$

where U is the internal energy, H the enthalpy, S the entropy, A the Helmholtz energy and G the Gibbs energy. In addition to its importance in physical chemistry, heat capacity is also a key property in engineering applications, where the design of typical processes involving heat transfer needs knowledge of this quantity.[49]

Isobaric heat capacities can be measured with any calorimeter that is able to measure heat flow and a change in temperature, although differential scanning calorimetry (DSC) is the most commonly used technique.[47] In practice, two types of ampules can be used: sealed ampules with a small vapor space or overflow ampules.[50] However, when the former is used, C_P is not measured directly as an increase in temperature of the saturated liquid at constant pressure will lead to complete evaporation.[48] Consequently, a quantity that is more closely related to this experimental measurement is the saturation heat capacity, C_{sat}, which measures the flow of heat needed to keep the liquid in a saturated state (q_σ) with a change in temperature. The definition of C_{sat} can be written analogously to Equations 1.7a and 1.7b as

$$C_{\mathrm{sat}} = \frac{đq_\sigma}{dT} = T\left(\frac{\partial S}{\partial T}\right)_\sigma, \tag{1.10}$$

and the relationship between C_{sat} and C_P can be obtained from the total differential of

$dS\,(T,P)$ and substituting Equations 1.8 and 1.10,

$$\begin{aligned} TdS &= T\left(\frac{\partial S}{\partial T}\right)_P dT + T\left(\frac{\partial S}{\partial P}\right)_T dP \\ \therefore\ TdS &= C_P\,dT - T\left(\frac{\partial V}{\partial T}\right)_P dP \\ \therefore\ C_{\text{sat}} &= C_P - T\left(\frac{\partial V}{\partial T}\right)_P \left(\frac{\partial P}{\partial T}\right)_\sigma , \end{aligned} \tag{1.11}$$

where the Maxwell relation $\left(\frac{\partial S}{\partial P}\right)_T = -\left(\frac{\partial V}{\partial T}\right)_P$ was used in the second line. In addition literature sometimes makes reference to another related heat capacity, C_σ, which is the change in enthalpy with temperature at constant pressure for a saturated liquid,[50]

$$C_\sigma = \left(\frac{\partial H}{\partial T}\right)_\sigma . \tag{1.12}$$

The relationship between C_σ and C_P can be obtained from the total differential of $dH\,(T,P)$ and substituting Equations 1.8 and 1.12, producing

$$\begin{aligned} dH &= \left(\frac{\partial H}{\partial T}\right)_P dT + \left(\frac{\partial H}{\partial P}\right)_T dP \\ \therefore\ C_\sigma &= C_P + \left(\frac{\partial H}{\partial P}\right)_T \left(\frac{\partial P}{\partial T}\right)_\sigma \\ \therefore\ C_\sigma &= C_P + \left[V - T\left(\frac{\partial V}{\partial T}\right)_P\right] \left(\frac{\partial P}{\partial T}\right)_\sigma . \end{aligned} \tag{1.13}$$

The last line follows from the fundamental equation for dH such that

$$\begin{aligned} dH &= TdS + VdP \\ \therefore\ \left(\frac{\partial H}{\partial P}\right)_T &= T\left(\frac{\partial S}{\partial P}\right)_T + V \\ &= V - T\left(\frac{\partial V}{\partial T}\right)_P . \end{aligned} \tag{1.14}$$

The difference between C_P and C_{sat} is often negligible at low vapour pressures and for

most liquids below their boiling point this difference is less than the typical uncertainty in high precision measurements.[47]

Lastly, the isochoric heat capacity can be difficult to measure directly with calorimetric methods, and is usually obtained indirectly from

$$C_P - C_V = TV\frac{\alpha_P^2}{\kappa_T}, \tag{1.15}$$

where $\alpha_P = \frac{1}{V}\left(\frac{\partial V}{\partial T}\right)_P$ is the isobaric expansion coefficient and $\kappa_T = -\frac{1}{V}\left(\frac{\partial V}{\partial P}\right)_T$ is the isothermal compressibility, which can both be determined experimentally from density measurements.[50]

1.3 Predicting Heat Capacities

Group Contribution methods (GCMs) form the most popular class of procedure used for the prediction of liquid-phase isobaric heat capacity, C_P,[51–54] having its origins in the pioneering work of Benson and co-workers.[55–57] However, with the development of cheminformatics and machine learning,[58] quantitative structure-property relationships (QSPRs) as well as other linear methods such as volume-based thermodynamics (VBT) and nonlinear regression methods have been widely used in prediction of physical properties, such as for ILs.[59]

This section presents a literature review of previous models used for the prediction of C_P for ILs. The predictive methods have been grouped as indicated in Figure 1.5 to provide structure to the discussion, as the majority of existing models can be grouped according to this scheme. It should be noted that the performance metrics stated here were taken directly from the original publications and are not directly comparable as the models differ in the composition of the data sets used and the treatment of the data during the training and testing phases. The lack of a standard evaluation platform for machine learning approaches has recently been addressed by Wu et al.[60] through the creation of MoleculeNet,

a benchmark collection for molecular machine learning; however, this benchmark does not include IL data, as required by these models.

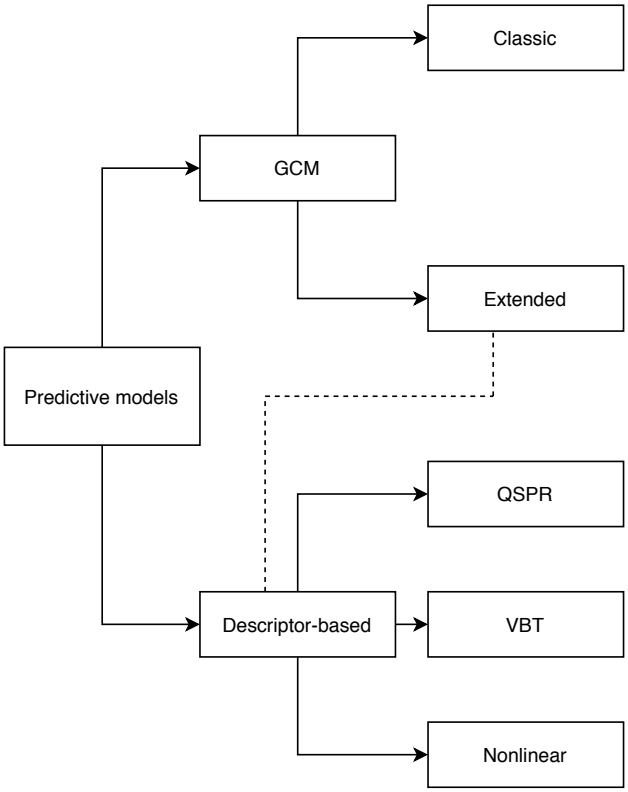
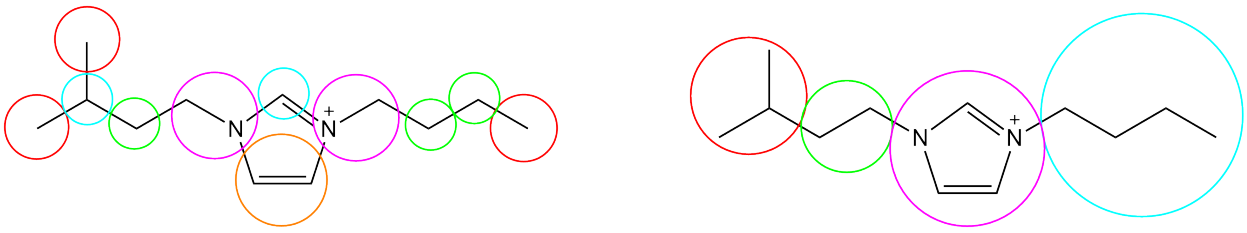


Fig. 1.5. Grouping of the predictive methods reviewed in this section.

1.3.1 Group Contribution Methods

The fundamental principle of GCMs is that physical properties are entirely dependent on the atoms/functional groups present in a molecule. To apply a GCM, a molecule is partitioned into atomic and/or functional groups, which are considered to contribute additively to a given physical property; in addition, these contributions stay fixed in other similar molecules.[28] Groups can be classified as zero-order, first-order or second-order. Zero-order groups are atoms only, and bonding between atoms is not considered. First-order groups consist of chemical functional groups, such as $-CH_3$ or $>C=C<$, in which the connectivity between atoms are included; however, this can still be limiting as it does

not consider the interaction between groups. Second-order groups remedy this by using first-order functional groups as the building blocks, and account for interactions between nearest neighbour groups, an example of such a group being $-C(CH_3)_3$.[55] The use of second-order groups is important when first-order groups do not give a sufficient description, although not all molecules will have second-order groups. As an example, the first-order and second-order groups present in the 3-butyl-1-isopentylimidazolium cation ($[C_4{}^iC_5im]^+$) are illustrated in Figure 1.6. Using first-order groups, all methyl groups in the cation are treated the same, while using a second-order description, the methyl groups are treated differently, based on the surrounding connectivity. GCMs are classified based on the type of groups that are used. Zero-order group contribution models are, however, not often encountered and the previous models presented here are classified as either first or second-order group contribution methods, based on the description given by the authors.



(a) First-order groups **(b)** Second-order groups

Fig. 1.6. Illustration of the first-order (a) and second-order (b) groups of the $[C_4{}^iC_5im]^+$ cation. In each case, different group types are indicated by circles of different colours.

GCMs can be used to either predict the ideal gas heat capacity, and then with the use of the corresponding states principle (CSP) determine C_P, or directly predict C_P.[51] The CSP approach, however, requires the prediction of both the ideal gas heat capacity as well as critical properties (i.e. T_c, V_c and P_c). One of the first GCMs for the estimation of critical properties was developed by Lydersen,[61] which paved the way to later model such as those of Joback,[62] Kleincewicz and Reid[63] and Constantinou and Gani,[64] among others. The Joback model further developed the model of Lydersen by using more functional groups and in later work, Joback used a GCM to model ideal gas heat capacities as a cubic function

of temperature.[52]

Růžička Jr. and Domalski presented one of the first to be widely employed GCM models for the direct prediction of C_P,[53,54] based on the model proposed by Benson and Buss,[55] which is one of the first temperature-dependent models. Temperature-dependent GCMs for C_P generally take the functional form of

$$C = A + BT + DT^2. \tag{1.16}$$

The parameters A, B, and D are determined as

$$\begin{aligned} A &= \sum_{i=1}^{k} n_i a_i, \\ B &= \sum_{i=1}^{k} n_i b_i \\ &\text{and} \\ D &= \sum_{i=1}^{k} n_i d_i, \end{aligned} \tag{1.17}$$

where n_i is the number of groups of type i, a_i, b_i and c_i are the parameters estimated for group i, and T is the temperature in Kelvin.[65] Where models differ from this form, the expressions used will be given.

1.3.1.1 First-Order Approaches

To the best of the authors' knowledge, Waliszewski et al.[66] were the first to use a group contribution method to predict IL heat capacities. They applied a first-order GCM, originally developed for molecular liquids,[67] to two ILs, [C_2C_1im][NTf_2] and [C_3C_1pyrr][NTf_2], at a single temperature of $293.15\,\mathrm{K}$, assuming that contributions from the cation and anion are additive. An error of approximately $12\,\%$ greater than experiment was reported.[66] A group additivity method based on the Joback CSP method,[62] was further developed by Ge

et al.[68] using the original parameter set of Joback extended to include $-SO_2-$, $-P-$ and $-B-$, allowing for the method to be applicable to a large range of ILs. The authors noted that the Joback model was developed for molecular liquids and thus no specific consideration is given to ionic charges in their model. This "extended" Joback method was used to first predict ideal gas heat capacity, C_P°, using

$$C_P^\circ = \left[\sum_k n_k A_{C_{Pk}} - 37.93\right] + \left[\sum_k n_k B_{C_{Pk}} 0.210\right] T + \left[\sum_k n_k C_{C_{Pk}} - 3.91 \times 10^{-4}\right] T^2 + \left[\sum_k n_k D_{C_{Pk}} - 2.06 \times 10^{-7}\right] T^3, \tag{1.18}$$

and then applying the CSP, C_P, is obtained from

$$\frac{C_P - C_P^\circ}{R} = 1.586 + \frac{0.49}{1 - T_r} + \omega \left[4.2775 + \frac{6.3(1 - T_r)^{\frac{1}{3}}}{T_r} + \frac{0.4355}{1 - T_r}\right]. \tag{1.19}$$

In the above equation, R is the gas constant, T_r is the reduced temperature, given by T/T_c where T_c is the critical temperature, and ω is the acentric factor, which is derived from the reduced vapour pressure and provides a measure of the sphericity of a molecule.[51] A data set of 961 points spanning 53 ILs was used to parameterise the model and the average absolute deviation (AARD)[I] across these points was $2.9\,\%$. Ge et al.[68] also applied the original Joback parameters to 15 halide-based ILs, and an AARD of $5.9\,\%$ resulted.

A problem with methods based on the corresponding states principle is that they also require the determination of critical properties, such as T_c and T_p, to obtain ω. Therefore, this model is not applicable if these properties are not known, or cannot be predicted with sufficient accuracy. As previously mentioned, prediction of normal boiling points and other critical properties is possible; however, this requires experimental measurements of

[I] $AARD\% = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100$, where y_i and $\hat{y}_i$ are the experimental and predicted values, respectively, for a set of n data points.

these properties, which is near-impossible as ILs decompose at temperatures approaching the normal boiling point, and this fact alone hinders the potential of this approach.[69] It should also be noted that this model was not tested on an external set—that being data points it was not trained with—and the error determined for the fitting is not necessarily representative of the predictive power. The model of Ge et al. was later extended by Gardas et al.[70] to include amino acid-based IL (AAILs) through the addition of parameters for N and P atoms bonded to four other atoms (>N< and >P< groups, respectively) as well as $-SO_2$ and $-SH$ groups.

Albert and Müller[71] used a similar first-order GCM, including a correction term to account for the type of ion and type of ring. The parameters were fit to a training set of 2210 data points from 86 ILs, and the model was then tested on 209 data points from 20 ILs, producing an AARD of $5.44\,\%$ when applied to the test set. The error here is larger than that of Ge et al. likely owing to the use of an external test set.

1.3.1.2 Second-Order Approaches

A second-order group additivity method was used by Gardas and Coutinho[72] with the general form of

$$C_P = R\left[A + B(\frac{T}{100}) + D(\frac{T}{100})^2\right], \tag{1.20}$$

making use of the approach of Růžička Jr. and Domalski, which is based on the GCM proposed by Benson and coworkers, extended such that C_P of a cyclic compound is the sum of contributions from acyclic compounds, and a correction for the type of cyclic compound accounting for ring constraints,[73] leading to subsequent deviations from the GCM proposed by Benson. Gardas and Coutinho developed the model using 12 groups consisting of three cation ring cores (1,3-dimethylimidazolium, 1-methylpyridinium, 1,1-dimethylpyrrolidinium), six whole anions, and three groups to describe substituents on the rings. Parameters were estimated using 2396 C_P data points spanning 19 ILs, resulting in an AARD of $0.36\,\%$ for the full fitting data set. It is important to note that no external

testing set was used to assess the prediction error, the data set used consisted of 1528 data points from a single IL ($[C_4C_1im][PF_6]$), and of the 19 ILs used, 14 were imidaozlium-based. Therefore, the errors might not necessarily be as low when applied to other classes of ILs.

As done similarly with the Ge model, Gardas et al.[70] also extended the work by Gardas and Coutinho to AAILs by including $[(C_1)_4N]^+$ and $[(C_1)_4P]^+$ groups, as well as parameters for seven amino acid anions. Soriano et al.[74] too developed a GCM influenced by the work of Gardas and Coutinho, but simplified such that C_P is taken as the sum of contributions from the full anion and full cation, rather than components of each. C_P of $[C_4C_1im]^+$ was calculated using the PM3 semi-empirical method and the usual statistical thermodynamics rigid-rotor harmonic-oscillator ideal gas approximations to the partition function,[75] and the contribution of other partner ions in the data set were calculated using this as a reference. Using this approach, the C_P contribution of a further 10 cations and 14 anions was calculated. The model was applied to a test set of 735 data points from 9 ILs and an AARD of $1.8\,\%$ was reported. This approach, of using each ion as a group, was also used in another GCM by Müller and Albert,[76] and was extended to 39 cations and 32 anions. The parameters were fitted using least-squares minimization of the difference between the experimental and predicted C_P, rather than using an *ab initio* or semi-empirically calculated reference C_P, as done by Soriano et al. A data set of 2443 points from 104 ILs was compiled, out of which 84 ILs were used for training and the remaining 20 for testing, producing an AARD of $4.4\,\%$ when applied to their test set. This methodology allows access to the C_P values of a total of 1248 pure ILs; however, it is limited to the ions in the fitting set that was used.

Nancarrow et al.[77] recognised the difficulty in comparing GCMs, since literature models are typically tested against different and varying data sets. In an attempt to address this, a data set of 2642 points spanning 96 ILs was compiled and the main GCMs were classified into two categories: (1) the "Lego" approach, which is based on small, simpler

functional groups and (2) the "Meccano" approach, based on larger, more complex groups, typically consisting of whole ions with limited functional groups. The Ge et al. model and Gardas and Coutinho model were used as examples of a Lego and Meccano models, respectively, and, since neither of these were parameterised for all functional groups in the full data set, were applied to subsets of the data set. This is a downfall of GCMs as they are not applicable to compounds containing groups that have not been parameterised. The Gardas method was applied to 2584 data points from 92 ILs and produced an AARD of $6.76\,\%$. However, when applied to a smaller data set of 1586 points spanning 45 ILs, constructed in a way such that both models can be applied to the same data, AARDs of $5.30\,\%$ and $5.54\,\%$ resulted for the Gardas (Meccano) and Ge (Lego) models, respectively. The authors noted that this is quite surprising, given that only three groups in the Lego model were parameterised specifically for ILs. It also provides valuable data showing that the increased flexibility and applicability of a first-order GCM does not lead to a significant loss in accuracy, compared to the specialised Gardas and Coutinho-type second-order methods. Since the Ge model uses a larger variety of smaller, more widely applicable groups, it was extended by adding $-$S$-$ and $-$SH functional groups. This model was then applied to the full data set, producing an AARD of $5.31\,\%$. It should be noted that the final model of Nancarrow et al. was tested on points it was parameterised with, and no external test set was used.

The most recent application of a second-order GCM was Chen et al.[78] This model is an extension of the method proposed by Gardas and Coutinho[72], to include more families of ILs and thus to allow for a more widely applicable model. The full data set included 3304 heat capacities from 61 ILs, parameterised with groups including six cation rings (ammonium, phosphonium, and piperidinium based cations were added), 14 anions (previously six), and the same three cation side chain groups. The model was trained using 44 ILs (2391 points) and then assessed on the remaining 913 points from 17 ILs, giving an AARD of $0.62\,\%$.

1.3.1.3 Extended Group Contribution Approaches

Valderrama and coworkers extended the classic GCM by appending additional terms to the functional form.[22,79] Initially, Valderrama and Rojas[79] put forward a model for the prediction of C_P expressed as

$$C_P = C_{P0} + \lambda \left[B(T - T_0) + D(T - T_0) \right]. \tag{1.21}$$

The reference heat capacity, C_{P0}, is an experimental value measured at a temperature T_0 of $298.15\,\mathrm{K}$. The parameters B and D are determined by fitting to experimental data and the *mass connectivity index (MCI)*, λ, is calculated for every IL using the functional groups present. The MCI is based on the concept of molecular connectivity as first introduced by Randić[80] and is calculated as the sum of the inverse of the mass connectivity interactions, which are the square root of the product of the masses, m,of two connected predefined groups, i and j,[79] determined using

$$\lambda = \sum \left(\frac{1}{\sqrt{m_i m_j}} \right)_{ij}. \tag{1.22}$$

In later work, Valderrama et al.[22] replaced the experimental C_{P0} with a GCM-calculated value at a fixed temperature of $T_0 = 298.15\,\mathrm{K}$. The groups used to parameterise the GCM portion of the model were the same as those used to determine the MCI. A data set of 469 C_P values from 32 ILs, and 126 points from 126 organic compounds was used for the parameterisation of the model, and 65 data points from 9 ILs were used for testing. This method produced an AARD of $2.6\,\%$ when applied to the test set.[22]

1.3.2 Quantitative Structure-Property Relationship Methods

GCMs predict properties as a weighted sum of contributions of predefined zero, first, or second-order groups within a molecule. Quantitative structure-property relationship

(QSPR) methodologies, on the other hand, predict macroscopic properties as a weighted sum of the contributions from various molecular descriptors. Molecular descriptors are defined by Todeschini and Consonni as "*the final result of a logical and mathematical procedure, which transforms chemical information encoded within a symbolic representation of a molecule into a useful number or the result of some standardized experiment.*"[81] There are a large amount of molecular descriptors that have been developed, and can be categorised according to the following: (i) **0D**-descriptors, which are dependent only on the atoms present, and not on the connectivity, and examples include atom counts, molecular weights, and partial charges; (ii) **1D**-descriptors, determined based on the structural representation, such as the functional groups present; (iii) **2D**-descriptors, providing topological representations of molecules, which consider the connectivity of the atoms and (iv) **3D**-descriptors, which use a geometrical representation of a molecule, considering the spatial arrangements of atoms. *Note that according to these definitions, zero-order, first-order and second-order GCMs can equally well be thought of as QSPR models featuring "pure" 0D-, 1D- or 2D-descriptors, respectively.*

Farahani et al.[82] used molecular parameters to develop a surprisingly simple correlation for the prediction of C_P using

$$\begin{aligned} C_P = &- 122.128.16826 + 0.45794\mathrm{T} + 12.8395\mathrm{N\_cation} - 56.85424\mathrm{CH3R\_cation} \\ &+ 19.25836\mathrm{N\_anion} - 11.36109\mathrm{nH\_anion}, \end{aligned} \tag{1.23}$$

where N_cation and N_anion are the cation and anion atomic counts, respectively, nH_anion is the number of hydrogen atoms present in the anion, CH3 R_cation is the number of methyl groups in the cation, and T is the temperature. A data set of 2940 experimental C_P values, spanning 56 ILs (consisting of 32 cations and 19 anions) was used, split into a training set of 2352 data points and a test set of 588 data points. When applied to the test set, the model produced an AARD of $2.51\,\%$.

Sattari et al.[83] developed two QSPR models, the first of which was developed using 3726 data points spanning 82 ILs, using 3001 points (61 ILs) for training and the remaining 725 points (21 ILs) for testing. Molecular descriptors were calculated and the genetic function approximation was used to select the most important descriptors. The final model contained 13 descriptors, 12 being binary combinations of the molecular descriptors and the other being a linear temperature term and the AARD of the test set was $2.32\,\%$. In the second model,[84] molecular parameters were determined and the genetic approximation function was used to reduce the number of descriptors to the most effective ones, and the final QSPR model was parameterised with these 14 atomic based groups. Parameters included the number of nitrogen, sulfur, chlorine, and non-hydrogen atoms as well as an overall atom count, and the number of triple bonds. A large portion of the parameters were based on the type of carbon in the alkyl chains, and the type of carbon in the aromatic rings. The model was trained on 2939 points from 65 ILs and then tested on 787 points from 17 ILs, for which an AARD of $1.65\,\%$ was reported.

Ahmadi et al.[85] developed the simple QSPR model,

$$\begin{aligned} C_P =& a_1 T^{a_2} + a_3 ln(T) + a_4 \mathrm{MW}^{a_5} + a_6 C_A + a_7 C_C + a_8 N + a_9 S + a_{10} O \\ &+ a_{11}(F + Br + Cl) + a_{12} B + a_{13}, \end{aligned} \tag{1.24}$$

where a_2 and a_5 are ≈ 1. This model uses temperature, T, molecular weight, W, the number of carbon atoms in the cation (C_C) and anion (C_A), as well as the total number of nitrogen (N), sulfur (S), oxygen (O), fluorine (F), bromine (Br), chlorine (Cl) and boron (B). The model was trained using 4072 heat capacities from 100 ILs, and tested on a set of 750 data points spanning 28 ILs, for which the AARD across the latter was $5.61\,\%$. The models produced by Sattari et al.[83] and Farahani et al.[82] were applied to this validation set, producing an AARD of $6.6\,\%$ and $14.2\,\%$, respectively. The errors produced by these models are much larger than those reported by the original authors, again highlighting the effect of the choice of test set used to assess a model. Paternò et al.[86] developed a

predictive model using *in silico* structural features determined with the VolSurf+ approach, which extracts the information contained in a molecular map of interaction energy, and transforms it in into a few interpretable numerical descriptors.[87] The Volsurf+ descriptors used consisted of 128 for the cation, accounting for features such as the cation core type and chain lengths, and 65 for the anion. These features were then compacted into five cation and four anion principle properties, which were used to develop a partial least squares regression model. The model was trained on data from 65 ILs, and tested on an external set of seven ILs for which the AARD produced was $5.07\,\%$

He et al.[88] proposed a QSPR model based on a series of *norm-indices*. The matrix norm-indices encode the molecular topology (a 2D representation using the connectivity and spatial distances) and other atomic descriptors (0D representations including the atomic weight, radius, electronegativity and atomic charge) of the ions to produce a large set of descriptors; the most significant of these (33) were then combined to give a second-order polynomial of temperature, with coefficients fitted for the prediction of C_P. An important distinction of this model from most other IL QSPR models is that the cation-anion interaction is taken into account through norm-indices which use the product of cation and anion quantities. The model was trained on 4610 data points from 148 unique ILs, and tested against 1354 data points for 37 unique species, giving an AARD on the latter set of $4.03\,\%$.

1.3.3 Volume-Based Thermodynamics

The foundations and proof-of-concept of volume-based thermodynamics (VBT) was originally formulated by the groups of Glasser and Jenkins in a series of papers from 1999 to 2005.[89–94] The basic principle of VBT is that thermodynamics properties of the condensed phase of ionic and ionic-covalent species, such as entropy and lattice potential energy or enthalpy, are correlated with the molecular formula unit volume, V_{m}.[91–93] It is important to note that V_{m} in many cases, such as lattice energies, is a simple and logical extension of

ionic radii, used in relationships such as the Kapustinskii equation. However, while ionic radii are only suitable for spherical ions, V_{m} is appropriate for more asymmetric ions, typically found in ILs. For example, it was shown that lattice energies correlate with $\sqrt[3]{V_{\mathrm{m}}}$;[94] this inverse relationship can be rationalized since smaller volumes suggests stronger interactions between ionic species. As noted by Glasser and Jenkins, an earlier form of the correlation between lattice enthalpy and volume with the same functional form, known as *Bartlett's rule*, was published by Mallouk et al.[95] On the other hand, entropies were shown to correlate linearly with the volume; again, this can be rationalized through noting that larger volumes will result in greater freedom of motion and thus a higher entropy (further rationalisation of this relationship is provided in the appendix of Reference 91) The formula unit volume of an ionic solid can be determined empirically as

$$V_{\mathrm{m}}/\mathrm{nm}^3 = \frac{V_{\mathrm{cell}}}{Z}, \tag{1.25}$$

where V_{cell} is the volume of the crystallographic unit cell and Z is the number of formula units in the cell. Alternatively, the V_{m} can also be determined from the molar volume as obtained using solid- (or liquid-) phase density measurements.[92] Through comparison of V_{m} of ionic species with a common cation or anion, single-ion volumes can be extracted. With the assumption of additivity of ionic volumes, V_{m} of any ionic species can then be determined from the independently established volumes of the constituent cation (V_{cation}) and anion (V_{anion}),[94,96,97]

$$V_{\mathrm{m}} = V_{\mathrm{cat}} + V_{\mathrm{an}}. \tag{1.26}$$

Although VBT was initially focused on the properties of ionic solids,[91,93] Glasser and Jenkins have also expanded their work to organic liquids[90] and ILs,[90,98] as defined in this work. Early progress in this area was reviewed in 2005 by the same authors,[89] with more recent additions, including progress in the area of ILs, discussed in a more recent review.[99] Of particular interest is Glasser and Jenkins's application of VBT to the isobaric heat capacities

of ILs, published in 2011.[98] The authors correlated V_{m} (obtained through liquid densities) of 13 imidazolium-based ionic liquids to C_P and determined a linear relationship (constrained through the origin) with $R^2 = 0.98$,

$$C_P = 1136.3 V_{\mathrm{m}} \tag{1.27}$$

The linear relationship existing between C_P and V_{m} for ILs was also independently pointed out earlier by Strechan et al.[100] and Gardas and Coutinho.[72] Paulechka et al.[101] later investigated this for 19 ILs consisting of imidazolium, ammonium, and pyrrolidinium-based ILs, from which it was concluded that the volumic heat capacity, C_P/V_{m}, at a fixed temperature of $298.15\,\mathrm{K}$ is constant within $\pm 5\,\%$,

$$\frac{C_P}{V_{\mathrm{m}}} = 1.95 \pm 0.02\,\mathrm{J/(K\,cm^3)}. \tag{1.28}$$

Independent of the progress made by the groups of Glasser and Jenkins using VBT, discussed above, the groups of Krossing and co-workers have also developed a range of volume-based correlations, specifically for ILs. In their first application, lattice free energies, $\Delta_{\mathrm{latt}}G$, calculated using VBT, were combined with solvation free energies, $\Delta_{\mathrm{solv}}G$, estimated using the COSMO solvation model,[102] to show that the standard free energies of fusion, $\Delta_{\mathrm{fus}}G^{\circ}$, of ILs are negative at room temperature; thus, lending thermodynamic support to the observed low melting points of these materials.[103] This work was followed by a volume-based predictive model in which the viscosity of a range of ILs was found to have an anion-dependent linear relationship with V_{m}, the latter determined from crystallographic data.[104] A departure from the need for experimentally determined volumes came through the work of Preiss et al., in which densities and isobaric heat capacities were correlated with molecular volumes.[21] In this work, multiple computational methods[II] were

II Methods used include: (i) Hofmann's simple incremental method.; (ii) Gavezzotti's method applied to gas-phase PM3, PM6, BP86/SV(P) and BP86/TZVP optimised structures and (iii) COSMO applied to the same structures as previously mentioned.

used to develop a linear correlation between calculated molecular volumes, V_{c}, and experimental single-ion volumes, V_{ion}, such that

$$V_{\mathrm{ion}} = aV_{\mathrm{c}} + b, \tag{1.29}$$

where a and b differ for the cation and anion. Equation 1.29 produced a correlation coefficient greater than 0.98 for all methods considered; however, volumes calculated using COSMO,[III] based on BP86/TZVP optimised structures, produced the most accurate correlation and were thus used to obtain V_{ion} throughout. Furthermore, it was pointed out that since C_P is dependent on entropy, it is expected to have a positive correlation with V_{m}, and as such C_P was modelled using

$$C_P = iV_{\mathrm{m}} + j, \tag{1.30}$$

where V_{m} is the sum of V_{cation} and V_{anion}, both determined using Equation 1.29. The regression was done using the C_P data of 17 ILs and the error assessed on a separate set of 17 ILs, at a fixed temperature. The training and test set were separated based on the water content, and the ILs with a low water content were used to train the model. This model was fit at both $298\,\mathrm{K}$ and $323\,\mathrm{K}$ producing an error of $6.0\,\%$ and $6.8\,\%$, respectively. Including information from quantum mechanical calculations in order to improve predictions using molecular volumes was again done in another contribution by Preiss et al.,[105] this time for the prediction of enthalpies of vaporization, $\Delta_{\mathrm{vap}}H$. The linear correlation between ${V_{\mathrm{m}}}^{2/3}$ and $\Delta_{\mathrm{vap}}H$, established earlier by Zaitsau et al.,[106] was improved by including the thermal correction to the gas phase electronic energy of the ions as an additional term in the multiple linear regression. The form of this correction can be obtained using standard statistical thermodynamics and includes contributions from the zero-point vibrational energy (ZPVE), as well as translational, rotational and vibrational degrees of freedom. In this work, the term "augmented VBT" (aVBT) was introduced to describe volume-based

III COSMO, implemented in TURBOMOLE, constructs a cavity based on Van der Waals radii, and the molecular volume is taken as the volume of the cavity.

correlations enhanced by quantum mechanical calculations.

Further application of aVBT by the group of Krossing produced prediction models for a range of IL properties including density, liquid entropy, viscosity, electric conductivity, static dielectric constant, transition enthalpies, melting points and critical micelle concentration, further demonstrating the vast predictive power of aVBT. All the developments coming from this work and references to the original literature describing these models can be found in the review by Beichel et al.[107]

1.3.4 Nonlinear Methods

The models presented thus far only consider linear correlations between C_P and the features used. It is possible that certain features can be used to describe C_P, but a nonlinearity needs to be considered. The models presented here make use of nonlinear machine learning methods.

Valderrama et al.[23] used the MCI to train an artificial neural network for the prediction of C_P. Temperature, the MCI of each ion, and the mass ratio of each ion were used as the input parameters. A feed-forward neural network with a single hidden layer consisting of 10 nodes was used. The network was trained using 477 data points from 31 ILs and tested on 65 data points from 9 ILs, which produced an AARD of $0.22\,\%$.

Zhao et al.[108] developed a single layer feed-forward neural network, employing the extreme learning machine (ELM) algorithm for the prediction of heat capacities, using σ-profiles (which quantify the electronic structure), molecular weight, and temperature as the input features. The full data set used consisted of 2416 data points from imidazolium, pyridinium, pyrrolidinium, and phosphonium-based ILs, and was split into a training set of 1933 data points and a test set of 483 data points. This model produced an AARD of $0.74\,\%$ when applied to the test set. Kang et al.[109] followed up on this by developing an ELM using the same data set as that of Zhao et al.,[108] replacing σ-profiles with electrostatic potential surface areas, a descriptor that discretises the electrostatic potential on the surface of a

molecule. The Multiwfn package was used to calculate 300 electrostatic potential surface areas (150 for each the cation and anion) for each IL in the data set. When applied to the test set, an AARD of $0.57\,\%$ resulted. Although models using multivariate linear regression (MLR) was also developed by Zhao et al.[108] and Kang et al.[109] using the same features, the ELMs performed significantly better: The Zhao and Kang linear models produced AARDs of $2.88\,\%$ and $2.51\,\%$, respectively, when applied to the test set. Therefore, focus is placed on the nonlinear models from these references. It should be noted that the training set used contains 1256 temperature-dependent data point from a single IL, and of the full 2416 data points used, 2203 are imidazolium-based ILs; the errors would likely not be as low when applied to a more diverse set of ILs. Barati-Harooni et al.[110] developed three models for the prediction of C_P, based on three machine learning algorithms. The first model, referred to as CSA-LSSVM, is based on the least squares support vector machine algorithm. The second model makes use of the Gene Expression Programming Algorithm (GEP Model), and the last model, termed the Hybrid-ANFIS model, is an adaptive neuro fuzzy interference system, for which the algorithm is a combination of that of fuzzy logic, and artificial neural networks. The models all used temperature and structural related parameters. A data set of 2940 points spanning 56 ILs was used, for which $80\,\%$ is used for training and the remaining $20\,\%$ is used to test the models. When applied to the test set, the CSA-LSSVM, GEP, and Hybrid-ANFIS models produced an AARD of $1.00\,\%$, $2.32\,\%$, and $1.83\,\%$, respectively.

The above models are summarised in Table 1.3. Again, it must be emphasised that the data sets used vary depending on the references and the AARD values should not be directly compared.

Table 1.3. Summary of previous prediction models reviewed here. Model type indicates the predictive method that was used. The number of data points and unique ILs given are the total number across both training and test sets.

Year	Reference	Model	Data points	Unique ILs	AARD%
2008	Ge et al.[68]	GCM	961	53	2.9
	Gardas and Coutinho[72]	GCM	2396	19	0.36
2009	Preiss et al.[21]	VBT	34	34	6.0 [a]
					6.8 [b]
2010	Soriano et al.[74]	GCM	3149	32	1.8
	Valderrama and Rojas[79]	GCM	469	32	2.6
2011	Valderrama et al.[23]	ANN	542	40	0.22
2013	Farahani et al.[82]	QSPR	2950	56	2.51
	Sattari et al.[83]	QSPR	3726	82	2.32
2014	Albert and Müller[71]	GCM	2419	106	5.44
	Sattari et al.[84]	GCM	3726	82	1.65
	Müller and Albert[76]	GCM	2443	104	4.4
2015	Nancarrow et al.[77]	GCM	2642	96	5.31
	Zhao et al.[108]	MLR	2416	46	2.88
		ELM	2416	46	0.74
2017	Barati-Harooni et al.[110]	SVM	2940	56	1.00
		GEP	2940	56	2.32
		ANN	2940	56	1.83
2018	Kang et al.[109]	MLR	2416	46	2.51
		ELM	2416	46	0.57
2019	Chen et al.[78]	GCM	3304	61	0.62
	He et al.[88]	QSPR	5964	185	4.03

[a] 298 K.
[b] 323 K.

1.4 Aims and Objectives

The main aim of the project was to apply supervised machine learning methods to develop generally applicable models for the prediction of the isobaric heat capacity of ionic liquids. The above aim included the following objectives:

1. Compilation of an extensive and diverse database of IL constant pressure heat capacities from the online ILThermo database. This involved structuring and "cleaning" the data.

2. Generation of low-energy ion structures for the ILs in the database, for which lin-

early independent features were calculated and used as descriptors in the machine learning models. Features were obtained from quantum mechanical and empirical calculations.

3. Development of a methodology to train and assess machine learning models. The machine learning methods were based on multiple linear regression and artificial neural networks. The models were developed to be generally applicable in that they can be applied to any novel IL, of which only the molecular structure is known.

Chapter 2

Theoretical Background

This chapter includes the theory that pertains to the thermodynamic property that is the focus of the work—the heat capacity, C_P , as well as the theoretical workings of the methods and calculations used throughout. Presented first is a thermodynamic explanation of C_P, including its fundamental definition and determination, as well as its relation to other thermodynamic properties. The second section presents the mathematics behind the machine learning methods used in this work. The final section gives a brief overview of the computational chemistry methodology, mostly density functional theory (DFT), employed in the molecular calculations.

2.1 Heat Capacity

Section 1.2 introduced the two commonly encountered liquid-phase heat capacities, C_P, and C_V, by definition and stated the relationship between them. The following gives a more in-depth breakdown of the thermodynamic origins, including a derivation of the relationship between the two.

2.1.1 Isochoric Heat Capacity

When a closed system experiences a changes in state, the change in internal energy, dU, is expressed using a total differential as

$$dU(T,V) = \left(\frac{\partial U}{\partial T}\right)_V dT + \left(\frac{\partial U}{\partial V}\right)_T dV. \tag{2.1}$$

Using the first law of thermodynamics ($dU = đq - P_{\text{ext}}dV$), the corresponding infinitesimal heat flow, $đq$, is thus given as

$$đq = \left(\frac{\partial U}{\partial T}\right)_V dT + \left[P_{\text{ext}} + \left(\frac{\partial U}{\partial V}\right)_T\right] dV, \tag{2.2}$$

which, at constant volume, simplifies to

$$đq_V = \left(\frac{\partial U}{\partial T}\right)_V dT. \tag{2.3}$$

The *isochoric heat capacity*, C_V, expresses the proportionality between heat flow and temperature change,

$$đq_V = C_V dT, \tag{2.4}$$

and, using Equation 2.3, can therefore be expressed as

$$C_V = \left(\frac{\partial U}{\partial T}\right)_V. \tag{2.5}$$

2.1.2 Isobaric Heat Capacity

The change in enthalpy, dH, can be written analogously to Equation 2.1 as

$$dH(T,P) = \left(\frac{\partial H}{\partial T}\right)_P dT + \left(\frac{\partial H}{\partial P}\right)_T dP. \tag{2.6}$$

It can be shown that if a reversible change of state occurs at constant pressure, then

$$dH = đq_P, \tag{2.7}$$

which allows for $đq_P$ to be expressed as

$$đq_P = \left(\frac{\partial H}{\partial T}\right)_P dT. \tag{2.8}$$

The *isobaric heat capacity*, C_P, is defined in a similar manner as C_V,

$$đq_P = C_P dT, \tag{2.9}$$

and using Equation 2.8, C_P is expressed as

$$C_P = \left(\frac{\partial H}{\partial T}\right)_P. \tag{2.10}$$

2.1.3 Relating Isobaric and Isochoric Heat Capacities

C_P and C_V are not equal since heat flow is a path dependent or an inexact differential, and the difference between these two quantities can be written as

$$C_P - C_V = \left(\frac{\partial H}{\partial T}\right)_p - \left(\frac{\partial U}{\partial T}\right)_V. \tag{2.11}$$

Since the enthalpy is defined as $H = U + PV$, the temperature derivative of dH at constant P can be expanded to give

$$C_P - C_V = \left(\frac{\partial U}{\partial T}\right)_p + P\left(\frac{\partial V}{\partial T}\right)_P - \left(\frac{\partial U}{\partial T}\right)_V. \tag{2.12}$$

Using Equation 2.1 and expressing the temperature derivative of internal energy along a path of constant pressure, it follows that

$$\left(\frac{\partial U}{\partial T}\right)_P = \left(\frac{\partial U}{\partial T}\right)_V + \left(\frac{\partial U}{\partial V}\right)_T \left(\frac{\partial V}{\partial T}\right)_P, \tag{2.13}$$

and substituting Equation 2.13 into Equation 2.12 gives

$$C_P - C_V = \left[\left(\frac{\partial U}{\partial V}\right)_T + P\right] \left(\frac{\partial V}{\partial T}\right)_P. \tag{2.14}$$

The fundamental equation for the change in internal energy is given as

$$dU = TdS - PdV, \tag{2.15}$$

and when divided by dV at constant T results in

$$\left(\frac{\partial U}{\partial V}\right)_T = T\left(\frac{\partial S}{\partial V}\right)_T - P. \tag{2.16}$$

Using one of the Maxwell relations, $\left(\frac{\partial S}{\partial V}\right)_T = \left(\frac{\partial P}{\partial T}\right)_V$, Equation 2.16 becomes

$$\left(\frac{\partial U}{\partial V}\right)_T = T\left(\frac{\partial P}{\partial T}\right)_V - P \tag{2.17}$$

and using this, the difference between C_P and C_V can be expressed as

$$C_P - C_V = T\left(\frac{\partial P}{\partial T}\right)_V \left(\frac{\partial V}{\partial T}\right)_P. \tag{2.18}$$

The relationship between the two quantities can therefore be expressed in terms of easily measurable state variables or quantities. Furthermore, two quantities that occur frequently in material science and are used to quantify the volumetric response of a system to a change in temperature or pressure are the *isobaric expansion*, α_P, and the *isothermal*

compressibility, κ_T, which are given by Equation 2.19 and Equation 2.20 respectively.

$$\alpha_p = \frac{1}{V}\left(\frac{\partial V}{\partial T}\right)_p \tag{2.19}$$

$$\kappa_T = -\frac{1}{V}\left(\frac{\partial V}{\partial P}\right)_T. \tag{2.20}$$

Using the cyclic rule to relate partial derivatives of P, V and T,[111] Equation 2.18 can be written as

$$C_P - C_V = -T\left(\frac{\partial V}{\partial T}\right)_P \left(\frac{\partial V}{\partial T}\right)_P \frac{1}{\left(\frac{\partial V}{\partial P}\right)_T}. \tag{2.21}$$

Consequently, the relationship between $C_P$and C_V can be expressed in terms of α_P and κ_T as

$$C_P - C_V = TV\frac{\alpha_P^2}{\kappa_T}. \tag{2.22}$$

2.1.4 Obtaining the Isobaric Heat Capacity

While Equation 2.22 can be used to determine C_P from a series of volume or density measurements, this first requires the determination of C_V. However, as stated in Section 1.2, C_V is difficult to measure, and is commonly obtained with the knowledge of C_P. Therefore, using Equation 2.22 to determine C_P becomes impractical. Fortunately, the need for C_V can be removed by introducing the *isoentropic compressibility*, κ_S, which is defined analogously to κ_T, but under conditions of constant entropy,

$$\kappa_S = -\frac{1}{V}\left(\frac{\partial V}{\partial P}\right)_S. \tag{2.23}$$

To develop the relationship of κ_S to Equation 2.22, one can again start with the fundamental equation for a change in internal energy ($dU = TdS - PdV$). At constant volume, it follows that

$$\left(\frac{\partial U}{\partial T}\right)_V = T\left(\frac{\partial S}{\partial T}\right)_V, \tag{2.24}$$

which allows for C_V to be expressed in terms of entropy as

$$C_V = T\left(\frac{\partial S}{\partial T}\right)_V. \tag{2.25}$$

The change in enthalpy is related to entropy through the fundamental equation for dH,

$$dH = TdS + VdP \tag{2.26}$$

and therefore, at constant pressure,

$$\left(\frac{\partial H}{\partial T}\right)_P = T\left(\frac{\partial S}{\partial T}\right)_P, \tag{2.27}$$

and consequently C_P can also be written in terms of entropy,

$$C_P = T\left(\frac{\partial S}{\partial T}\right)_P. \tag{2.28}$$

The *adiabatic index* can now be defined as the ratio of the two heat capacities,

$$\gamma = \frac{C_P}{C_V} \tag{2.29}$$

which using Equation 2.25 and Equation 2.28 can be expressed as

$$\gamma = \frac{\left(\frac{\partial S}{\partial T}\right)_P}{\left(\frac{\partial S}{\partial T}\right)_V}. \tag{2.30}$$

The cyclic rule can then be used to write the isobaric change in entropy with a change in temperature as

$$\left(\frac{\partial S}{\partial T}\right)_P = -\frac{\left(\frac{\partial P}{\partial T}\right)_S}{\left(\frac{\partial P}{\partial S}\right)_T}, \tag{2.31}$$

and the isochoric change as

$$\left(\frac{\partial S}{\partial T}\right)_V = -\frac{\left(\frac{\partial V}{\partial T}\right)_S}{\left(\frac{\partial V}{\partial S}\right)_T}. \tag{2.32}$$

Finally, substituting Equation 2.31 and Equation 2.32 into Equation 2.30 results in

$$\frac{C_P}{C_V} = \frac{\kappa_T}{\kappa_S}, \tag{2.33}$$

which can be substituted into Equation 2.22 to give

$$C_P = \frac{TV\alpha_p^2}{\kappa_T - \kappa_S}. \tag{2.34}$$

Equation 2.34 provides an indirect method to measure C_P, rather than by direct calorimetric means (see Equation 1.10). Lastly, isoentropic measurements are not practical, and κ_S is typically obtained through measurement of the speed of sound (υ) and density (ρ),[112] as

$$\kappa_S = \frac{1}{\rho}\frac{1}{\upsilon^2}. \tag{2.35}$$

However, the above method is not ideal as the denominator in Equation 2.34 is often very small at moderate pressures.[113]

2.1.5 Isobaric Heat Capacity and Temperature Dependence of State Functions

The change in thermodynamic state functions resulting from a change in temperature, such as the change in molar enthalpy, ΔH, and the change in molar entropy, ΔS, can be determined using C_P. If the enthalpy, H, is known at a reference temperature T_1, it can be extrapolated to T_2 by integrating Equation 2.10 to give

$$\Delta H = \int_{T_1}^{T_2} C_P dT, \tag{2.36}$$

and if C_P is taken to be constant or the temperature interval is small, this results in

$$H(T_2) = H(T_1) + C_P(T_2 - T_1). \tag{2.37}$$

Otherwise, or if higher accuracy is required, it is typical to express C_P as a power series,

$$C_P(T) = \alpha + \beta T + \gamma T^2, \tag{2.38}$$

resulting in the integrated expression,

$$H(T_2) = H(T_1) + \alpha\left(T_2 - T_1\right) + \frac{\beta}{2}\left(T_2^2 - T_1^2\right) + \frac{\gamma}{3}\left(T_2^3 - T_1^3\right). \tag{2.39}$$

Likewise, the change in entropy, ΔS, can be expressed by integrating Equation 2.28 to obtain

$$\Delta S = \int_{T_1}^{T_2} \frac{C_P(T)}{T} dT. \tag{2.40}$$

As done for enthalpy above, C_P can again be expressed as a polynomial in T and the integration done analytically.

2.2 Machine Learning Theory

This section presents the theoretical background behind the machine learning methods that are used throughout this work, including linear and regularised regression, and artificial neural networks. The theory presented here is collated from *Hands-on Machine Learning with Scikit-Learn, Keras & TensorFlow* by Aurélien Géron,[114] and *Pattern Recognition and Machine Learning* by Christopher M. Bishop.[115]

2.2.1 Linear Regression

Linear regression is among the simplest machine-learning methodologies, with the fundamental idea being that a linear relationship exists between the features and property that is being modelled. Linear regression makes predictions as a sum of weighted input features and a bias term such that

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n, \tag{2.41}$$

where $\hat{y}$ is the predicted value, x_i is the i^{th} input feature with a weight β_i, β_0 is the bias term, and n is the number of features. This can be written in a vectorised form as

$$\hat{y} = \boldsymbol{\beta} \cdot \boldsymbol{x}, \tag{2.42}$$

in which $\boldsymbol{\beta}$ is the weight vector containing β_0 to β_n, and $\boldsymbol{x}$ is the feature vector containing x_0 to x_n ($x_0 = 1$).

Determining the coefficients is referred to as fitting the model. A linear regression model is fit by finding the values of $\boldsymbol{\beta}$ that produces the lowest error, which is done by minimizing the mean squared error (MSE),

$$\begin{aligned} \text{MSE}(\boldsymbol{\beta}) &= \frac{1}{m} \sum_{i=1}^{m} \left(\hat{y}_i - y_i\right)^2 \\ &= \frac{1}{m} \sum_{i=1}^{m} \left(\boldsymbol{\beta}^T \boldsymbol{x}_i - y_i\right)^2, \end{aligned} \tag{2.43}$$

where y_i is the target value of $\hat{y}_i$ for m observations. The MSE can also be referred to as the *cost function*, which is how the error is assessed.

The value of $\boldsymbol{\beta}$ that minimises Equation 2.43 can be found using the *normal equation*,

$$\hat{\boldsymbol{\beta}} = \left(\boldsymbol{X}^T \boldsymbol{X}\right)^{-1} \boldsymbol{X}^T \boldsymbol{y} \tag{2.44}$$

where $\hat{\boldsymbol{\beta}}$ is the value of $\boldsymbol{\beta}$ that minimizes the cost function, $\boldsymbol{X}$ is the feature vectors of the training set, and $\boldsymbol{y}$ is the target vector containing y_1 to y_n. Solving the normal equation becomes very slow with a larger number of features, as this increases the computational complexity, and a faster approach to finding β is by using *gradient descent*.

2.2.1.1 Gradient Descent

With gradient descent, the parameters in $\boldsymbol{\beta}$ are iteratively adjusted to minimise the MSE rather than using the closed-form solution in Equation 2.44. $\boldsymbol{\beta}$ is randomly initialised, the MSE is then calculated, and the value of $\boldsymbol{\beta}$ is updated until the cost function reaches a minimum. Each weight update is referred to as a step, and a minimum is determined by evaluating the gradient of the cost function, with respect to each weight, β_i, at each step, such that

$$\begin{aligned}\nabla_{\boldsymbol{\beta}}\mathrm{MSE}(\boldsymbol{\beta}) &= \begin{pmatrix} \frac{\partial}{\partial\beta_0}\mathrm{MSE}(\boldsymbol{\beta}) \\ \frac{\partial}{\partial\beta_1}\mathrm{MSE}(\boldsymbol{\beta}) \\ \vdots \\ \frac{\partial}{\partial\beta_n}\mathrm{MSE}(\boldsymbol{\beta}) \end{pmatrix} \\ &= \frac{2}{n}(\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y})\boldsymbol{X}^T.\end{aligned} \tag{2.45}$$

Once the gradient has been calculated, $\boldsymbol{\beta}$ is updated using

$$\boldsymbol{\beta}^{\tau+1} = \boldsymbol{\beta}^{\tau} - \eta\nabla_{\boldsymbol{\beta}}\mathrm{MSE}(\boldsymbol{\beta})^{\tau} \tag{2.46}$$

where τ is the step number, and η is the *learning rate*, which determines the size of the step. Using Equation 2.45 requires a calculation involving the whole batch of training data and is referred to as *batch gradient descent*, which can become slow. *stochastic gradient descent* offers a faster implementation by calculating the gradient for only a single, random instance at every step. Gradient descent is preferred over the normal equation when there

is a large number of features, as it decreases computational complexity. However, $\boldsymbol{\beta}$ is solved using the normal equation in Scikit-learn.

2.2.2 Regularised Linear Regression

When developing a linear-regression model, if too few input features are used, the variance in data is often not well described, resulting in underfitting, meaning that neither the training data or new data is modelled well. This can be rectified with the inclusion of more features; however, too many features reduces the degrees of freedom in a model, which can lead to overfitting of the data, in which training data is well described, but the model is not flexible to describe new data. Overfitting can be reduced with regularisation methods, which constrain the weights of less important features, allowing for a more flexible model. Three popular methods of regularisation are Lasso Regression, Ridge Regression, and Elastic Net Regression.

2.2.2.1 Lasso Regression

Least Absolute Shrinkage and Selection Operator (Lasso) regression[116] constrains the weights by attaching a regularisation term to the MSE, creating a new cost function, $J(\boldsymbol{\beta})$, such that

$$J(\boldsymbol{\beta}) = \text{MSE}(\boldsymbol{\beta}) + \alpha \sum_{i=1}^{n} |\beta_i| \tag{2.47}$$

where α is the *regularisation coefficient* that determines how much the weights are constrained, chosen so as to minimize the cost function, and determined using a technique such as cross-validation, and $|\beta_i|$ is the l_1 norm of the weight vector.[I] $\boldsymbol{\beta}$ is again determined by minimizing the cost function, $J(\boldsymbol{\beta})$; however, the cost function becomes nondifferentiable when any coefficient β_i is equal to zero. Consequently, gradient descent can still be used to find the coefficients, but a subgradient vector, $\boldsymbol{g}$, is used when any one of the

I The l_n norm of a vector $\boldsymbol{w}$ with m elements is defined as $||\boldsymbol{w}||_n = \sqrt[n]{|w_1|^n + |w_2|^n + |w_3|^n + ... + |w_m|^n}$.

β_i values is zero. This subgradient vector is an intermediate vector that is between the gradient vectors around the point where $\beta_i = 0$, determined by

$$\boldsymbol{g}(\boldsymbol{\beta}) = \nabla_{\boldsymbol{\beta}}\mathrm{MSE}(\boldsymbol{\beta}) + \alpha \begin{pmatrix} \mathrm{sign}(\beta_1) \\ \mathrm{sign}(\beta_2) \\ \vdots \\ \mathrm{sign}(\beta_n) \end{pmatrix} \quad \text{, where} \quad \mathrm{sign}(\beta_i) = \begin{cases} -1, & \text{if } \beta_i < 0 \\ 0, & \text{if } \beta_i = 0 \\ 1, & \text{if } \beta_i > 0 \end{cases}. \tag{2.48}$$

Using the above, the gradient can be solved at every point and $\boldsymbol{\beta}$ can be updated as in Equation 2.46.

2.2.2.2 Ridge Regression

As with Lasso regression, *Ridge regression*[117] constrains weights by adding a regularisation term to the cost function; however, the regularisation term uses half of the square of the l_2 norm of the weight vector, such that

$$J(\boldsymbol{\beta}) = \mathrm{MSE}(\boldsymbol{\beta}) + \alpha\frac{1}{2}\sum_{i=1}^{n}\beta_i^2 \tag{2.49}$$

While Lasso regression can result in certain features being completely dropped, Ridge regression will retain all features.

2.2.2.3 Elastic Net Regression

Lasso Regression is preferred over Ridge Regression when feature reduction is required; however, the former can often result in the over-reduction of features. *Elastic Net* is a mixture of Lasso and Ridge regression, allowing for a more controlled reduction of features. The cost function is defined as

$$J(\boldsymbol{\beta}) = \mathrm{MSE}(\boldsymbol{\beta}) + r\alpha\sum_{i=1}^{n}|\beta_i| + \frac{1-r}{2}\alpha\sum_{i=1}^{n}\beta_i^2 \tag{2.50}$$

where r controls the mix ratio, which is how much of each regularisation is used.

2.2.3 Artificial Neural Networks

A neural network consists of three types of layers: (i) the *input layer*; (ii) the *hidden layers* and (iii) the *output layer*. Each layer contains a finite number of nodes and a bias node(not present in the output layer), and every node in a given layer is connected to every node in the next layer. Each node in the input layer is an input feature, and each node in the output layer is a dimension of the predicted value, and the nodes in the hidden layers are where computation happens. In the discussion below, a neural network with a single hidden layer, and a single node in the output layer will be considered, but the results can be generalized to any number of layers. This general network architecture is shown in Figure 2.1.

First, each input feature x_i is passed into an input node, z_i. Then, each node in the hidden layer is determined as a weighted sum of the input nodes, z_i–z_n, plus a bias term, b_1, referred to as an *activation*, a_j, and is calculated as

$$a_j = \sum_{i=1}^{n} (w_{ij} z_i + b_1). \tag{2.51}$$

Each value of a_j is then transformed using a differentiable *activation function*, ϕ, introducing nonlinearity, to give the input node for the next layer, z_j,

$$z_j = \phi(a_j). \tag{2.52}$$

Once z_j is calculated, this is passed to the next layer (the output layer), for which the activation a_k is calculated as

$$a_k = \sum_{j=1}^{m} (w_{jk} z_j + b_2). \tag{2.53}$$

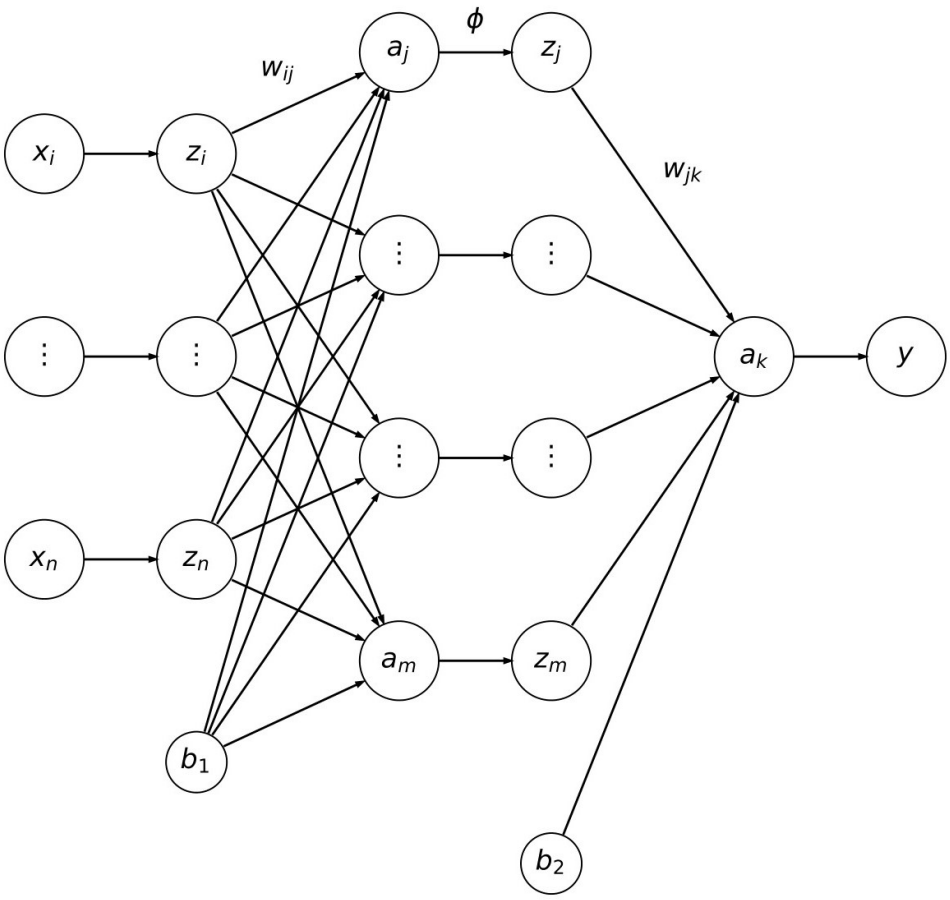


Fig. 2.1. General architecture of a single-layer feed-forward neural network where the input features, x_i–x_n, are passed into the input layer, z_i–z_n. The hidden layer consists of nodes a_j–a_m, passed through the activation function, ϕ, producing z_j–z_m. The node a_k represents the output layer, giving the predicted value, y. Input and hidden layers connected by weights Wij, hidden and output layers connected by weights W_{jk}. b_1 and b_2 are the bias nodes for the input and hidden layers, respectively.

For regression problems, no activation function is applied to the output layer;[II] therefore, a_k is simply the predicted value, $\hat{y}$, which can be written as the overall network function,

$$\hat{y} = \sum_{j=1}^{m} \left(w_{jk} \phi \sum_{i=1}^{n} (w_{ij} z_i + b_1) + b_2 \right). \tag{2.54}$$

The bias terms, b_1 and b_2, can be absorbed into the vector of weights by including an input

II For classification problems it is typical to apply an activation function that transforms the output to either 0 or 1.

feature $x_0 = 1$ and Equation 2.54 becomes

$$\hat{y} = \sum_{j=0}^{m} \left(w_{jk} \phi \sum_{i=0}^{n} w_{ij} z_i \right). \tag{2.55}$$

The above is referred to as forward propagation of information through the network, and such a network is called a *feed-forward neural network (FFNN)*, usually trained using gradient descent to determine the weights. As described in Section 2.2.1.1, this requires minimisation of the cost function with respect to each weight. FFNNs use the *backpropagation* algorithm to evaluate these gradients.

2.2.3.1 Backpropagation

As with linear regression, the cost function used is the mean square error (MSE)

$$\mathrm{MSE}(\boldsymbol{w}) = \frac{1}{r} \sum_{l=1}^{r} (\hat{y}_l - y_l)^2 \tag{2.56}$$

where $\boldsymbol{w}$ is the weight vector containing all weights, r is the number of data points, and y_l is the target value of $\hat{y}_l$. For simplicity, $\mathrm{MSE}(\boldsymbol{w})$ will henceforth be written as E. Gradient descent requires the partial derivative of E with respect to each weight. For the output layer, this is written as

$$\frac{\partial E}{\partial w_{jk}} = \frac{2}{r} \sum_{l=1}^{r} (\hat{y}_l - y_l) z_j. \tag{2.57}$$

The $\frac{2}{r}$ factor is a constant, and can be written as c, and the above is then written as

$$\frac{\partial E}{\partial w_{jk}} = c \sum_{l=1}^{r} (\hat{y}_l - y_l) z_j. \tag{2.58}$$

Now, considering the hidden layer, the partial derivative of E with respect to a weight, w_{ij}, can be expressed using the chain rule as

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial a_j}\frac{\partial a_j}{\partial w_{ij}}. \tag{2.59}$$

The error of a node j, δ_j, is defined as

$$\delta_j = \frac{\partial E}{\partial a_j} \tag{2.60}$$

and from Equation 2.51 it follows that

$$\frac{\partial a_j}{\partial w_{ij}} = z_i. \tag{2.61}$$

Consequently, Equation 2.59 can be expressed as

$$\frac{\partial E}{\partial w_{ij}} = \delta_j z_i. \tag{2.62}$$

The error of the node k, δ_k, is the error of the single output node, written as

$$\delta_k = c\sum_{l=1}^{r}(\hat{y}_l - y_l). \tag{2.63}$$

Likewise, δ_j can be written as

$$\delta_j = \frac{\partial E}{\partial a_j} = \frac{\partial E}{\partial a_k}\frac{\partial a_k}{\partial a_j} \tag{2.64}$$

and the following can be obtained:

$$\delta_j = \delta_k z_i w_{jk}. \tag{2.65}$$

As δ_k can be readily calculated, δ_j can be obtained for all hidden nodes by recursively applying Equation 2.65, and then Equation 2.62 can be used to obtain partial derivatives

of E with respect to each weight. These partial derivatives then lead to $\nabla_{\boldsymbol{w}}\mathrm{MSE}(\boldsymbol{w})$ and then gradient descent is applied as described previously to update the weights as

$$\boldsymbol{w}^{\tau+1} = \boldsymbol{w}^{\tau} - \eta\nabla_{\boldsymbol{w}}\mathrm{MSE}(\boldsymbol{w})^{\tau}, \tag{2.66}$$

where τ is the step of the gradient descent, and η is the learning rate.

2.2.3.2 Activation Functions

Backpropagation requires an activation function, ϕ, that is differentiable, as this is required to learn weights using gradient descent. While ϕ can be linear, a nonlinear ϕ introduces nonlinearity between the input and hidden layers, allowing for neural networks to solve complex problems. Typical such activation functions include:

1. Sigmoid:
$$\phi(a_j) = \frac{1}{1+e^{-a_j}} \tag{2.67}$$

2. Hyperbolic tangent (tanh):
$$\phi(a_j) = \frac{e^{a_j} - e^{-a_j}}{e^{a_j} + e^{-a_j}} \tag{2.68}$$

3. Rectified Linear Unit (ReLU):
$$\phi(a_j) = \mathsf{max}(0, a_j) = \begin{cases} 0 & \text{for } a_j < 0 \\ a_j & \text{for } a_j \geq 0 \end{cases} \tag{2.69}$$

4. Softmax
$$\phi(a_j) = \frac{e^{a_j}}{\sum_{j=0}^{m} e^{a_j}} \tag{2.70}$$

2.2.3.3 Optimisers

The weights of a neural network are trained, or optimised, using gradient descent. While training can be slow, this can be sped up by using a faster optimiser (still based on gradient descent), such as the following:

1. Stochastic Gradient Descent (SGD)

   Gradient descent computes the gradient using every training instance at every step. SGD picks a random instance at every step, and the gradient is calculated for that instance only.

2. Adaptive Gradient Algorithm (Adagrad)

   The Adagrad algorithm first calculates the vector of squared gradients, $\boldsymbol{s}$, which is used to update the weight vector,

   $$\begin{aligned} &1.\ \boldsymbol{s} \leftarrow \boldsymbol{s} + \nabla_{\boldsymbol{w}}\mathrm{MSE}(\boldsymbol{w}) \otimes \nabla_{\boldsymbol{w}}\mathrm{MSE}(\boldsymbol{w}) \\ &2.\ \boldsymbol{w} \leftarrow \boldsymbol{w} - \eta\nabla_{\boldsymbol{w}}\mathrm{MSE}(\boldsymbol{w}) \oslash \sqrt{\boldsymbol{s} + \epsilon}, \end{aligned} \tag{2.71}$$

   where ϵ is a smoothing term (typically set to $\epsilon = 10^{-10}$) to ensure the radicand is nonzero, and $\otimes$ and $\oslash$ are the elemental multiplication and division, respectively. The result is an adaptive learning rate, which is scaled based on the steepness of the gradient of the current step.

3. Root Mean Square Propagation (RMSProp)

   The RMSProp algorithm is as follows:

   $$\begin{aligned} &1.\ \boldsymbol{s} \leftarrow \gamma\boldsymbol{s} + (1-\gamma)\nabla_{\boldsymbol{w}}\mathrm{MSE}(\boldsymbol{w}) \otimes \nabla_{\boldsymbol{w}}\mathrm{MSE}(\boldsymbol{w}) \\ &2.\ \boldsymbol{w} \leftarrow \boldsymbol{w} - \eta\nabla_{\boldsymbol{w}}\mathrm{MSE}(\boldsymbol{w}) \oslash \sqrt{\boldsymbol{s} + \epsilon} \end{aligned} \tag{2.72}$$

   where γ is the decay rate, typically set to $\gamma = 0.9$.

4. Adaptive moment estimation (Adam)

The Adam algorithm is as follows:

$$
\begin{aligned}
&1.\ \boldsymbol{m} \leftarrow \gamma_1 \boldsymbol{m} - (1-\gamma_1)\nabla_{\boldsymbol{w}}\mathrm{MSE}(\boldsymbol{w}) \\
&2.\ \boldsymbol{s} \leftarrow \gamma_2 \boldsymbol{s} + (1-\gamma_2)\nabla_{\boldsymbol{w}}\mathrm{MSE}(\boldsymbol{w}) \otimes \nabla_{\boldsymbol{w}}\mathrm{MSE}(\boldsymbol{w}) \\
&3.\ \hat{\boldsymbol{m}} \leftarrow \frac{\boldsymbol{m}}{1-\gamma_1^t} \\
&4.\ \hat{\boldsymbol{s}} \leftarrow \frac{\boldsymbol{s}}{1-\gamma_2^t} \\
&5.\ \boldsymbol{w} \leftarrow \boldsymbol{w} + \eta \hat{\boldsymbol{m}} \oslash \sqrt{\hat{\boldsymbol{s}} + \epsilon}
\end{aligned}
\tag{2.73}
$$

where γ_1 and γ_2 are the momentum decay and scaling hyperparameters respectively, and t is the iteration number.

2.3 QM Theory

This section briefly discusses theory of a general nature for which individual references will not be given, but rather a general reference to the textbook which provided the source material for the section: *Introduction to Computational Chemistry* by Frank Jensen.[118]

2.3.1 Hartree-Fock Theory

Atoms and molecules demonstrate wave-particle duality and can therefore be represented with a wavefunction, Ψ, for which the energy, E, can be determined from the time-independent Schrödinger Equation,

$$\hat{H}\Psi = E\Psi. \tag{2.74}$$

$\hat{H}$ is the Hamiltonian operator, which is the sum of the kinetic and potential energy operators. To obtain the Hamiltonian, the kinetic energy of all nuclei and all electrons must be considered, as well as the attraction between nuclei and electrons, and repulsion be-

tween electrons. The Born-Oppenheimer approximation is applied, which treats all nuclei as stationary relative to the electrons, having zero kinetic energy, thus transforming Equation 2.74 into the electronic Schrödinger Equation,

$$\hat{H}_e \Psi_e(\boldsymbol{r}, \boldsymbol{R}) = E_e \Psi_e(\boldsymbol{r}, \boldsymbol{R}). \tag{2.75}$$

Here, $\boldsymbol{r}$ and $\boldsymbol{R}$ are the vectors containing the electronic and nuclei coordinates, respectively, and $\hat{H}_e$ is the electronic Hamiltonian, expressed as

$$\hat{H}_e = -\frac{1}{2}\sum_i \nabla_i^2 - \sum_{i,A} \frac{Z_A}{|r_i - R_A|} + \sum_{i<j} \frac{1}{|r_i - r_j|}, \tag{2.76}$$

where ∇_i^2 is the Laplacian operator of electron i, Z_A is the charge of nucleus A, and $r_{i,A}$ is the distance between electron i and nucleus A; all quantities are expressed in atomic units.

The operator must then be applied to a suitable wave function. For molecules, it is customary to construct the wave function from *spin orbitals*, where each is a combination of a spatial orbital and a suitable spin function. The ground state wave function, Ψ_0, for a system with N electrons and N spin orbitals, can then be built from a *Slater determinant*,

$$\Psi_0 = \frac{1}{\sqrt{N}} \begin{vmatrix} \psi_1(r_1) & \psi_1(r_2) & \cdots & \psi_1(r_N) \\ \psi_2(r_1) & \psi_2(r_2) & \cdots & \psi_2(r_N) \\ \vdots & \vdots & \ddots & \vdots \\ \psi_N(r_1) & \psi_N(r_2) & \cdots & \psi_N(r_N) \end{vmatrix} \tag{2.77}$$

in which each column of the determinant contains the wave function, describing the *molecular orbitals (MOs)* for a single electron i, ψ_i. The Slater determinant fulfils the antisymmetry requirement of a wave function, such that it adheres to the *Pauli antisymmetry principle*. This principle states that the wave function for fermions (particles of half-integer

spin) must change sign when the spatial and spin coordinates of any two particles are exchanged. As a consequence, no two electrons can have the same four quantum numbers and two electrons in the same orbital must have opposite spins. This requirement is explicitly satisfied using a determinant, since the exchange of two rows or columns changes the sign of a determinant, and the duplication of rows or columns results in a determinant that is zero. The energy of the Slater determinant is written as

$$E = \sum_{i=1}^{N} h_i + \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} (J_{ij} - K_{ij}) \tag{2.78}$$

where h_i is the kinetic and potential energy of electron i, and J_{ij} and K_{ij} are the *Coulomb* and *exchange* integrals, respectively.

Next, it is necessary to determine the MOs that produce a minimum energy, E, achieved by applying the *variational theorem* to Equation 2.78. The variational theorem states that the energy of any approximate wavefunction is greater than or equal to the exact energy. The energies will only be equal if the wavefunction is the exact function, occurring when the approximate wavefunction energy is at a minimum. Therefore, the "best" function is generated by constructing a trial wavefunction containing one or more parameters, and minimizing the energy as a function of these parameters. This produces the Hartree-Fock equations for an electron in MO i,

$$\hat{F}_i \psi_i = \epsilon_i \psi_i. \tag{2.79}$$

The *Fock operator*, $\hat{F}_i$, consists of the kinetic and potential energies of the electron ($\hat{h_i}$), and its interaction with all other electrons, through the Coulomb and exchange operators, $\hat{J_{ij}}$ and $\hat{K_{ij}}$, respectively; expressed as

$$\hat{F}_i = \hat{h}_i + \sum_{j}^{N} (\hat{J_{ij}} - \hat{K_{ij}}). \tag{2.80}$$

The MO ψ_i can be expressed as a linear combination of atomic orbitals (LCAO) such that

$$\psi_i = \sum_k c_{ik}\chi_k \tag{2.81}$$

where χ_k is an atomic orbital with an expansion coefficient, c_{ik}. The collection of atomic orbitals is referred to as a *basis set*. Substituting the basis set expansion into the Hartree-Fock equations gives the Roothaan-Hall equations, given here in matrix form,

$$\boldsymbol{FC} = \boldsymbol{SC\epsilon} \tag{2.82}$$

in which $\boldsymbol{F}$ is the Fock matrix, $\boldsymbol{C}$ is the matrix containing the MO coefficients, $\boldsymbol{S}$ is the overlap matrix, and $\boldsymbol{\epsilon}$ is the diagonal matrix of orbital energies, which are solved using the *self consistent field* (SCF) procedure. The SCF procedure is an iterative procedure used to calculate the Hartree-Fock energy, E. This is done by obtaining an initial guess for the coefficients c_{ik} which are used to obtain the initial Fock matrix $\boldsymbol{F}$. This matrix is then diagonalized and the resulting coefficients are then used to calculate a new Fock matrix. This is repeated until the coefficients used to construct the "newest" Fock matrix are equal to those resulting from the diagonalization.

2.3.2 Density Functional Theory

The problem with HF theory is that electrons are treated as independent particles, experiencing an average interaction with the other electrons in the system, resulting in an error in the calculated energy as there is no instantaneous coupling (correlation) considered between the electrons' motions. The difference between the HF energy and the exact energy is referred to as the *correlation energy*, which can be determined with methods such as Møller-Plesset perturbation theory, Configuration Interaction (CI) and Coupled Cluster (CC);[119] however, *Density Functional Theory (DFT)* can be used to address this deficiency,

at least in part, at a much lower computational cost. Two theorems put forward by Hohenberg and Kohn [120] form the basis of DFT. The first proved that a unique relationship exists between the electron density, ρ, that depends only on three spatial coordinates, and the electronic energy. The second theorem proved that the ground-state electron density minimizes the energy functional, $E[\rho]$, which can be expressed in terms of three component functionals as

$$E[\rho] = T[\rho] + E_{en}[\rho] + E_{ee}[\rho] \tag{2.83}$$

in which $T[\rho]$ is the kinetic energy functional, $E_{en}[\rho]$ gives the energy due to nuclei-electron attraction, and $E_{ee}[\rho]$ yields the energy resulting from electron-electron repulsion, which contains the Coulomb and exchange functionals, $J[\rho]$ and $K[\rho]$, respectively. $E_{en}[\rho]$ and $J[\rho]$ can be solved numerically using the electron density at a point r, $\rho(r)$ as

$$E_{en}[\rho] = \sum_a \int \frac{Z_a \rho(r)}{|r - r'|} dr \tag{2.84}$$

and

$$J[\rho] = \frac{1}{2} \int \int \frac{\rho(r)\rho(r')}{|r - r'|} dr dr'. \tag{2.85}$$

$T[\rho]$ forms part of the energy functional known as Thomas-Fermi (TF) theory, $E_{TF}[\rho]$,

$$E_{TF}[\rho] = T[\rho] + E_{en}[\rho] + J[\rho] \tag{2.86}$$

and the inclusion of $K[\rho]$, associated with Dirac, forms the Thomas-Fermi-Dirac (TFD) model, a forerunner to DFT theory. An issue with both TF and TFD theories is that the assumption of a uniform, non-interacting, single-electron gas is made, which is not sufficient to describe molecular systems.

Kohn and Sham (KS) suggested the introduction of MOs, to improve on the description of these expressions. The KS model is based on the construction of a reference system of non-interacting electrons, for which the MOs result in a wavefunction giving the same

electron density as the real system. The kinetic energy and Coulomb energies of electrons in the reference system can be determined, using the reference system MOs, with the wave-based expressions in HF theory. In HF, kinetic energies are calculated using the assumption of non-interacting electrons; however, as this is not the case, the KS model splits the actual $T[\rho]$ into two contributions—$T_{KS}[\rho]$, and a small correction, which can be thought of as a "kinetic correlation energy" $T_{KS}[\rho]$ is then calculated exactly (under the assumption of non-interacting electrons) and the remaining kinetic energy forms part of an exchange-correlation energy, $E_{xc}[\rho]$. Using the KS approach, the DFT energy is expressed as

$$E_{DFT}[\rho] = T_{KS}[\rho] + E_{en}[\rho] + J[\rho] + E_{xc}[\rho] \tag{2.87}$$

where

$$E_{xc}[\rho] = (T[\rho] - T_{KS}[\rho]) + (E_{ee}[\rho] - J[\rho]) \tag{2.88}$$

and $T_{KS}[\rho]$ is obtained from a wavefunction that consists of MOs in a Slater determinant. The exchange-correlation energy thus includes the correction in kinetic energy between the real and reference system, as well as the difference between the real electron-electron repulsion (that includes the effects of electron correlation and exchange) and that calculated from the reference system. At the heart of DFT is the construction of the exchange-correlation functional, for which various approximations and procedures exist. Further discussion of this lies outside the scope of this work, but a full taxonomy of popular DFT functionals and their performance can be found in the recent work of Mardirossian and Head-Gordon.[121]

Chapter 3

Methods and Model Development

This chapter describes the aspects involved in the development of the models presented in this work. First, a detailed explanation of the data set determination is given, including the procedure used to clean and prepare the raw data. The next section details the generation of the features used, including molecular volumes, electrostatic potential-based features, and other molecular descriptors, which have been chosen based on the performance in previous models. An assessment of the calculation procedures of molecular volumes, as well as temperature-dependence of the data set is also presented in this section. The assessment metrics used are then presented, which include standard metrics such as the average absolute relative deviation (AARD) and coefficient of determination (R^2).

This work makes use of both linear and nonlinear regression methods. The former includes multiple linear regression and regularised regression, and the nonlinear regression employed here is a feed forward artificial neural network (FFNN); the general fitting procedure is presented for each method. First, linear regression is addressed, where the treatment of the training and test set split of the data, as well as the feature reduction procedure, are outlined. This is followed by the fitting procedure of the FFNN, explaining the determination of the train:test split, and the optimisation of the network architecture and hyperparameters. *Note that theoretical details of the regression methods used are given*

in Chapter 2.

The last section in this chapter describes the external data set. As the fitting procedures differ between the linear and nonlinear models, the external data set is used to assess the final models when applied to a set of data, not present in any training set.

3.1 Data Collection and Preparation

Building a predictive model begins with determining a data set, which are the data points that will be used to train the model. As the aim of this project is to predict the isobaric molar heat capacities of liquid-phase ILs, C_P , experimental values of C_P for these species are used to train the models. While there is a large amount of literature present in which IL C_P values have been measured, searching through literature to obtain these values can become tedious. ILThermo[122,123] is an online database containing many experimentally determined properties of ILs, and was used to obtain experimental values of C_P.[101] The pyILT2 Python library is used to automate the search and extraction of data on ILThermo, limiting the search to constant-pressure heat capacities for pure compounds in the liquid phase. The quality of the experimental data is important and in addition to the intrinsic and human error of measurement, the impact of impurities should be considered. A source of error in experimental measurement of C_P is impurities, which can have an effect of the same order as the uncertainty in the measurement[124] and impurities such as water can significantly increase the value of C_P .[101] Although ILThermo does not report the purity of samples; aspects of the data cleaning can help to reduce the effect of this error. 8591 C_P values at various temperatures were extracted from ILThermo, and the data was cleaned according to the following criteria:

1. Although all values of C_P returned by the search were measured at a constant pressure, this pressure differed between references; that being, some references measured C_P values at $100\,\mathrm{kPa}$ and others at $101.3\,\mathrm{kPa}$. As the pressure needed to be

consistent across the data set, only points measured at $101.3\,\mathrm{kPa}$ were used[I] and all other points were removed.

2. Cross-checking with the original literature revealed that some data points were mislabelled as liquid-phase on ILThermo where in fact, these were measured below the glass transition point. These data points were initially identified through large jumps in C_P values with an increase in temperature and such points were consequently removed.

3. If an experimental value of C_P was reported by more than one reference for a certain IL, the data from only one source was kept. The reference used was chosen based on:

   (a) The year of the reference; assuming that measurements carried out in more recent years would have used more accurate methods.

   (b) The experimental error reported for each reference on ILThermo; favouring a lower error.

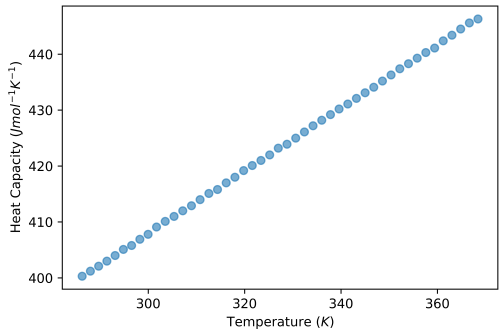
   (c) How many temperature-dependent data points each reference contained; favouring references which measured a larger amount of data points.

4. If there were multiple values for C_P at a single temperature recorded by a single reference, the average of these values was used to minimize experimental errors such as those possibly brought about by the presence of impurities.

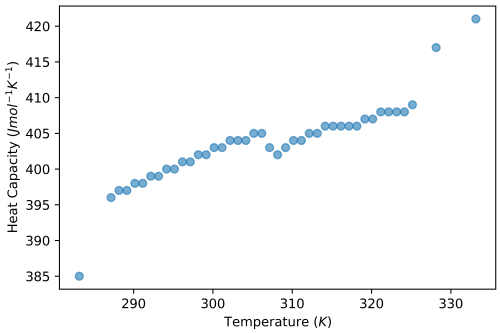
5. ILs for which C_P did not follow a reasonable temperature dependence were removed from the data set. This was determined by plotting the temperature dependence for each IL, and those that showed an irregular temperature dependence, as illustrated for 2-hydroxy-N-methylethaniminium pentanoate in Figure 3.1, were removed. If there was only one value of C_P for an IL, it is assumed to follow a reasonable temperature dependence and all single-temperature C_P values are kept in the data set.

I While standard pressure (100 kPa) is preferred, more data points were available at atmospheric pressure (101.3 kPa) and therefore the latter was used.

At this point, these plots were only used to identify irregularities; however, the temperature dependence of each IL, and the data set as a whole, are assessed in Sections 3.2.3 and 4.1.1.2, respectively. The reduced list of ILs, prior to assessment of temperature dependence, can be found in the electronic supporting information (ESI) at `https://bit.ly/tw-esi`.



(a) 1-butyl-3-methylimidazolium hexafluorophosphate



(b) 2-hydroxy-N-methylethaniminium pentanoate

Fig. 3.1. Temperature dependence of C_P illustrated for two ILs (a) with a clear and consistent (b) with an inconsistent temperature dependence, showing a discontinuity around 305 K. ILs showing an irregular temperature dependence as in (b) were removed from the data set.

After removing points with an irregular temperature dependence, the final data set contained 2463 data points, spanning 208 unique ILs containing 105 unique cations, distributed into 5 cation families,[II] and 51 unique anions. Table 3.1 summarizes the final data set, and Figure 3.2 shows the distribution of data points among the cation families. The full final data set can be found in the ESI.

3.2 Descriptors

Once a data set has been obtained, the next step in building a model is determining the descriptors that will be used, as the predictive power of a model is highly dependent on

II Although more cation families did show a regular temperature dependence, these were removed to form part of an external set, addressed in Section 3.6.

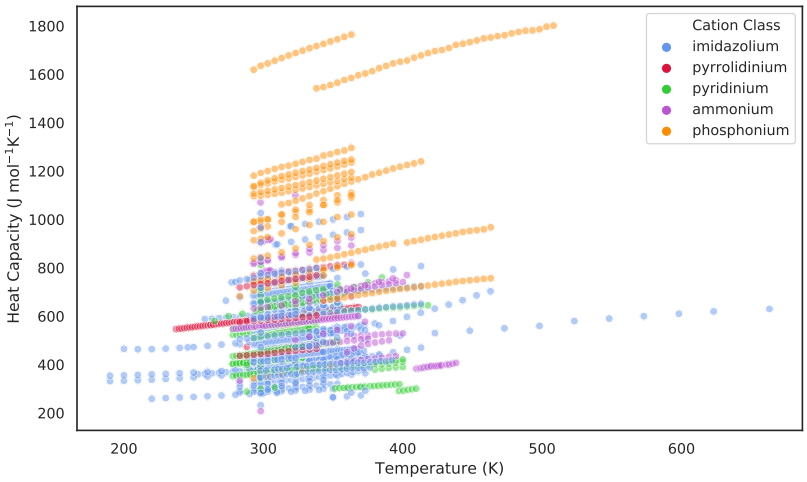


Fig. 3.2. Distribution of data points by cation class.

Table 3.1. Summary of the data set used in this work, classed based on the cation core.[a]

Cation class	Temperature range	C_P range	Data points	Unique ILs
Imidazolium	190.0–663.1	84.0–1027.0	1279	112
Pyridinium	265.04–418.15	288.0–811.0	464	46
Ammonium	278.32–470.0	207.47–1100.0	209	20
Phosphonium	283.15–513.15	343.0–1806.31	300	20
Pyrrolidinium	237.44–368.4	424.0–812.0	211	10

[a] Heat capacities in J/(mol K) and temperatures in K.

the choice of features. This section describes all features that are used, as well as how these are calculated.

3.2.1 Structure Generation

All feature calculations were done for the anions and cations separately. Low-energy conformers were generated using the simulated annealing method implemented in the xTB program.[125] The lowest energy conformer generated with the simulated annealing was then optimised with the PBE generalized gradient approximation (GGA) functional[126] and def2-TZVP triple-ζ basis set.[127] PBE is the preferred non-empirical GGA functional recommended by Perdew and co-workers.[128] This is further supported by the benchmark study of

isomerization energies of organic molecules by Grimme et al.[129] in which the use of a GGA functional, in particular PBE, combined with at least a triple-ζ basis set, is recommended for geometry optimization. The RI approximation[130] was used with the def2/J auxiliary basis set[131] using the ORCA 4.0 software.[132] The structures were optimised with a tight convergence criteria. All subsequent feature calculations were done using these optimised structures.

3.2.2 Molecular Volumes

Glasser and coworkers[90,133] showed that a simple linear correlation exists between molecular volume, V_{m}, and standard entropy and as a thermodynamic relationship exists between latter and C_P, a relationship exists between V_{m} and C_P. V_{m} has been successfully used for the prediction of C_P (see Section 1.3.3) and based on this, is used as a descriptor in this work.

V_{m} can be determined experimentally: The volume of the unit cell, V_{cell}, as well as the number of formula units in the cell, Z, can be determined the from the X-ray crystal structure, and using these quantities V_{m} is calculated as

$$V_{\mathrm{m}}/\mathrm{nm}^3 = \frac{V_{\mathrm{cell}}}{Z}. \tag{1.25}$$

However, this does rely on the experimental quantities stated above, which would not be readily available for novel, yet to be synthesised ILs. Three methods used for the calculation of V_{m} are explored in this work, each discussed in detail below.

3.2.2.1 Solvent Excluded Surface Volumes

The GEPOL algorithm[134] was used to compute molecular surfaces, from which volumes can be obtained. Three types of surfaces can typically be defined:(i) a Van der Waals surface; (ii) a solvent accessible surface (SAS) and (iii) a solvent excluded surface (SES). The

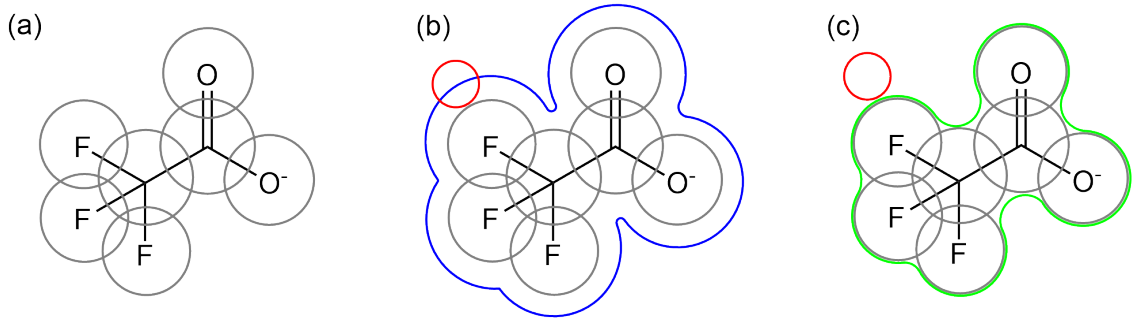


Fig. 3.3. Illustration of GEPOL calculated surfaces: (a) The Van der Waals volume created by Van der Waals spheres, indicated in grey; (b) The SAS surface, indicated in blue; (c) The SES surface, indicated in green. The solvent probe molecule is represented in red for both (b) and (c).

Van der Waals surface is the result of the interlocking atomic Van der Waals spheres of a molecule. To determine the SAS, a Van der Waals sphere is placed over each atom in the molecule and, based on the defined size of a solvent molecule, the spaces between spheres are filled with new spheres to build a solvent accessible surface. Here the solvent is considered a sphere that rolls along the Van der Waals surface and the SAS is the resulting surface generated from the path of the centre of the solvent molecule. This creates an envelope for which surface area and volume can be calculated. The SES is determined in the same manner as the SAS, but the surface is determined from the contact point between the solvent sphere and the Van der Waals spheres, rather than the centre of the solvent sphere.[134] These three surfaces are illustrated in Figure 3.3 for the trifluoroacetate anion. The GEPOL algorithm is implemented in codes that calculate implicit solvation free energies. Consequently, V_{m} is calculated using the SES, referred to as ${V_{\mathrm{m}}}^{SES}$, extracted from a conductor-like polarizable continuum model (CPCM)[135] calculation with the dielectric constant set to infinity ($\epsilon_{\mathrm{r}} = \infty$) and a solvent sphere radius of $1.3\,\text{Å}$.[III]

III The SES and specified solvent radius were used as these are the default settings applied when the CPCM solvation model is implemented.

3.2.2.2 Isodensity Surface Volumes

Vertices are generated to represent an isosurface for which the enclosed volume can be calculated. The vertices are placed at an isodensity surface where the electron density, $\rho(r)$, is $0.001\, e\, {a_0}^3$ as this contains at least 96% of the electronic charge.[136] This method is implemented in the Multiwfn package,[137,138] and molecular volumes calculated using the electron density isosurface are denoted as $V_{\mathrm{m}}^{\mathrm{ISO}}$.

3.2.2.3 Atomic and Bond Contribution Volumes

Zhao et al.[139] developed a procedure to calculate Van der Waals volumes that does not use the three dimensional molecular structure or connectivity, based on a sum of Bondi volumes over all atoms with empirical corrections for the number of bonds and rings, calculated using

$$V_m^{\mathrm{ABC}}(\text{Å}) = \sum V_{\mathrm{atom}} - \sum(5.92N_{\mathrm{B}} - 14.7R_{\mathrm{A}} - 3.8R_{\mathrm{NA}}), \tag{3.1}$$

where V_{atom} is the atomic volume of each atom, N_{B} is the number of bonds present, and R_{A} and R_{NA} are the number of aromatic and non-aromatic rings present, respectively. Molecular volumes calculated using the atomic bond contribution are denoted as $V_{\mathrm{m}}^{\mathrm{ABC}}$, and are calculated with the Mordred descriptor calculator.[140] As explained in Section 1.3.3, V_{m} of each species is the summation of the volumes of the component ions, calculated here with each of the above methods. The correlation between these volumes were plotted in Figure 3.4 and although a correlation between the calculated volumes is clear, the differences in the correlation coefficients presented indicate that these volumes do not provide the same description of the data, requiring assessment of the accuracy of each.

3.2.3 Temperature

The temperature dependence needs to be described for the data set as a whole, but as this differs for each species, it was necessary to determine whether each IL can be described by the same function of temperature. Of the 208 ILs in the data set, 43 had only a single C_P value and the temperature dependence could not be assessed. A linear temperature dependence,

$$C_P = a + bT, \tag{3.2}$$

was assessed for the remaining 164 ILs and the R^2 values for each of these fittings were calculated to determine how well the relationship describes each species; a histogram of these values is presented in Figure 3.5. Equation 3.2 describe $77.4\,\%$ of these ILs with an $R^2 \geq 0.99$, indicating that though it is typical to describe C_P with a polynomial function of temperature,[101] the majority of the data set is sufficiently described by a linear temperature dependence.

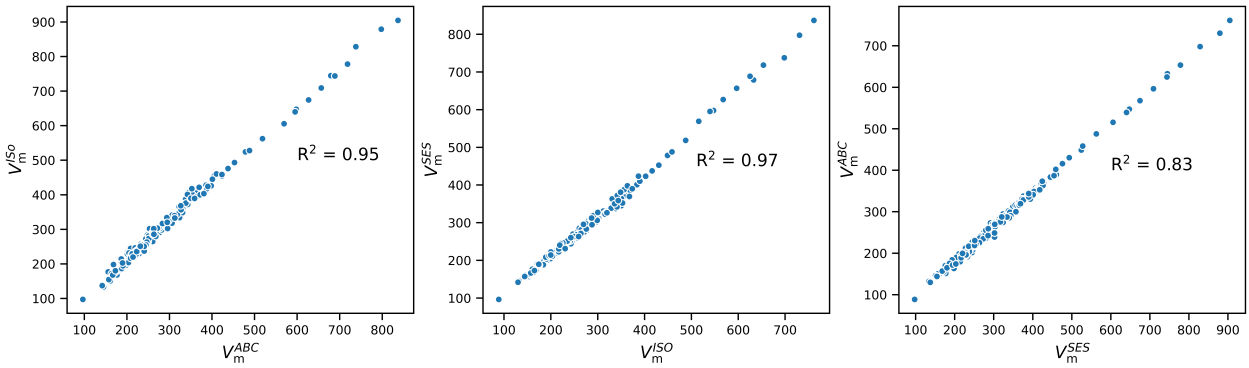


Fig. 3.4. Correlations plots of $V_{\mathrm{m}}{}^{ABC}$, $V_{\mathrm{m}}{}^{ISO}$ and $V_{\mathrm{m}}{}^{SES}$. All volumes are given in $\mathrm{Å}^3$.

3.2.4 Electrostatic Potential-based features

The electrostatic potential, $V(r)$, can be calculated at a point r as

$$V(r) = \sum_A \frac{Z_A}{|R_A - r|} - \int \frac{\rho(r')}{|r' - r|} dr', \tag{3.3}$$

where the sum is over all nuclei, Z_A is the nuclear charge located at R_A and $\rho(r')$ is the electronic charge density at the point r'. This property can be calculated at any point in space around a molecule, and has been used before as a fundamental descriptor of a molecular system.[141] $V(r)$ can be related to many molecular properties such as energies and noncovalent interactions.[142] To provide a qualitative view of the positive and negative regions of a molecule or ion, $V(r)$ is typically mapped onto the surface of a molecule, often defined by a Van der Waals envelope, which can be constructed from the charge distribution as an isosurface, once again using $\rho(r) = 0.001\, e\, {a_0}^3$.

$V(r)$, mapped onto an isodensity surface, was calculated for the optimised structures of the anions and cations, separately, using the Multiwfn package.[137,138] The electrostatic potential was subsequently used in two ways as features in this work. The first was as electrostatic potential surface areas, which provide a way to discretise this continuous property, and the second way was as general interaction properties functions, which provide statistical measures of the electrostatic potential within a molecule.

3.2.4.1 Electrostatic Potential Surface Areas

The use of the electrostatic potential surface area, S_{EP}, was inspired by the work of Kang et al.,[109] and the S_{EP} feature was determined for each IL as follows: $V(r)$ is calculated and mapped onto the isodensity surface of each ion. A $V(r)$ range of $-627.6\,\mathrm{kJ/mol}$ to $627.6\,\mathrm{kJ/mol}$ was then split into 300 equal bins and the total surface area of the ion is binned according to the value of $V(r)$ at the midpoint of the bin. This $V(r)$ range was chosen based on the work of Kang et al.[109] in which a $V(r)$ range of $\pm 150\,\mathrm{kcal/mol}$

(ie. $\pm 627.6\,\mathrm{kJ/mol}$) was used. Each of these bins was then treated as a descriptor, referred to as an electrostatic potential surface area element, $S_{EP(i)}$. Typical histograms, as represented by the 1-ethyl-3-methylimidazolium cation and alkylsulfate anions, are shown in Figure 3.6. Here, it is illustrated that the electrostatic potential bins associated with the alkyl chain are consistent across the series, but by increasing the length of the alkyl chain, the surface area associated with those value increases, and as a result, the bins increase in size. Figure 3.6 also demonstrates that since the S_{EP} features are calculated for each ion separately, altering structural elements of one ion has no effect on the S_{EP} features of the other component ion.

3.2.4.2 General Interaction Properties Functions

Politzer and coworkers[143,144] introduced the concept of a General Interaction Properties Function (GIPF) formed from a set of statistically well-defined measures of the electrostatic; these features have also been found to correlate well with the strength of electrostatic interactions between molecules, which is of course the dominant noncovalent interaction in ILs.[145] The first of these features is the variance in $V(r)$ over m points on the surface of a molecule, σ^2, which is calculated as

$$\sigma^2 = \frac{1}{m}\sum_{i=1}^{m}\left[V(r_i) - \bar{V}_s\right]^2, \tag{3.4}$$

which requires $\bar{V}_s$, the average electrostatic potential on the surface, calculated using

$$\bar{V}_s = \frac{1}{m}\sum_{i=1}^{m} V(r_i). \tag{3.5}$$

σ^2 is calculated for both the cation, σ_+^2, and the anion, σ_-^2, which have been found to correlate with electrostatic interaction tendencies.[145] The second GIPF feature explored is

the average deviation of $V(r)$ on the surface, Π, which is calculated using

$$\Pi = \frac{1}{m} \sum_{i=1}^{m} \left| V(r_i) - \bar{V}_s \right| . \tag{3.6}$$

Π can be used as an indication of the local polarity of a molecule,[145] and is once again calculated for both the cation, Π_+, and the anion, Π_-. While σ^2 and Π are similar in that both contain the deviation from the mean, σ^2_- will be more sensitive to positive and negative extrema, as it is the square of the deviation.

3.2.5 Molecular Descriptors

QSPR models determine the relationship between a desired property and the molecular structure, which requires a way to describe the molecular structure in quantitative terms; molecular descriptors, as defined in Section 1.3.2, can be used to achieve this quantification. Molecular descriptors such as atomic counts, chain lengths, atomic charges, and cation core types, have been successful in building QSPR models for C_P prediction[82–84,86,88] and therefore, were used here. Molecular descriptors are appealing for predictive models as there are a large number of open-source software packages available that can be used to calculate these descriptors, such as RDKit, PyChem, and Mordred.[140,146,147] The Mordred descriptor calculator was chosen for this work as it is freely available as a Python package and is capable of calculating a larger number of features compared to other freely available software. This package can calculate 1825 molecular descriptors, including, 0D, 1D, 2D, and 3D-descriptors, as described in Section 1.3.2; the full list of descriptors can be found at `https://bit.ly/mordred_descriptors`.

3.2.6 Feature Scaling

Regularised linear regression and a FFNN were used in this work, both sensitive to the scale of the input features, as coefficient sizes are dependent on the feature size. Therefore, each

element x_i of an input feature $\boldsymbol{x}$ was scaled by transforming it into x_i',

$$x_i' = \frac{x_i - \bar{x}}{\sigma}, \tag{3.7}$$

where $\bar{x}$ is the mean value and σ is the standard deviation of the feature. This ensures that all features have a mean value of zero and a standard deviation of one. The scaling of features also allows for direct interpretation of the coefficients, that being, the importance of a feature will be indicated by the absolute value of the coefficient.[114]

3.3 Model Assessment

The models presented here were compared using the following standard criteria:

1. The coefficient of determination, R^2, calculated as

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}, \tag{3.8}$$

   where y_i is the observed value, $\hat{y}_i$ is the predicted value, and $\bar{y}$ is the mean of the observed values. R^2 is used to indicate how well the model describes the variance in the data. Generally, the inclusion of more features, regardless of the descriptive quality, will better describe the variance in the data resulting in a higher value of R^2. Therefore, to allow for a fair comparison between models which use a different number of features, the adjusted R^2 is used.

2. The adjusted R^2, R^2_{adj}, is calculated as

$$R^2_{adj} = 1 - \left[\frac{(1 - R^2)(n-1)}{n-k-1}\right], \tag{3.9}$$

   where n is the number of data points and k is the number of features used in the model.

3. The Average Absolute Relative Deviation (AARD) is the average relative deviation across all points, determined as

$$\text{AARD}\% = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100. \tag{3.10}$$

3.4 Linear Regression Model Development

This section describes the steps involved in training the linear regression models (see Section 4.1 for results), beginning with the treatment of the data with respect to the training and testing split, followed by the procedure used for feature reduction, if it was required. The regression methods used here are implemented in the Scikit-learn Python library.[148]

3.4.1 Train:Test Split

Developing a model requires two non-overlapping sets of data: a training set, which is used to train the model; and a test set, which is used to assess the model performance when applied to points it was not trained with. Two factors were considered when making this split. First, it is important that points used for the test set are never in the training set, as the model cannot be assessed accurately if it is tested on data it has already "seen". As the data set consists of several ILs for which C_P has been measured across multiple temperatures, the models need to be capable of predicting the temperature dependence of each IL. This means that temperature-dependent points of a given IL cannot be split across the training and test set, as then the model would have essentially been trained with the partial temperature dependence of these species. For this reason, when a train:test split was made, the temperature-dependent points of a single IL were kept together in either the training or test set.

It was initially decided that the data should be split such that $90\,\%$ of the ILs are in the

training set and the remaining $10\,\%$ are used for the test set.[IV] However, this then leads to the second factor that needs to be considered—the performance of the model can be influenced based on how the data is split. Therefore, instead of using a single train:test split, 1000 different 90:10 splits of the data were made and a linear model was trained for each. This resulted in 1000 independently trained models over which the average performance can be assessed and the effectiveness of this approach was investigated in Section 4.4. It should be noted that the 1000 splits were only made once and all models were trained using the same splits.

3.4.2 Feature reduction

When training a model it is important to consider how many features are used. Too few features can result in underfitting of the data and conversely, too many features can result in overfitting of the data. Overfitting can be avoided by reducing the number of features such that only features which are highly correlated with the predicted property are used. Here, the feature space was reduced using Least Absolute Shrinkage and Selection Operator (Lasso) Regression. Lasso regression adds a regularisation term to the cost function, constraining the weights of unimportant features. Lasso regression is used over other regularisation methods, as weights of unimportant features can be reduced to zero, resulting in automatic feature selection.[114] When using Lasso Regression, the coefficients, $\boldsymbol{\beta}$, for n features are solved by minimising the following cost function,

$$J(\boldsymbol{\beta}) = \mathrm{MSE}(\boldsymbol{\beta}) + \alpha \sum_{i=1}^{n} |\beta_i| \,, \tag{2.47}$$

which requires the regularisation coefficient, α, responsible for constraining the weights of the features (details are given in Section 2.2.2). Consequently , for each of the 1000 splits, a range of 300 α values between 10^{-2} and 10^{2} were assessed using 5-fold cross-validation.

IV While an 80:20 train:test split is more commonly made (see Section 1.3), the fitting procedure used here was inspired by the work of Beckner et al.[149] in which a 90:10 split is made.

In cross-validation, the ILs in the training set are split into an 80:20 train:validation set five times, and each time the training data is fit using each value of α for which the error is determined using the validation set, and the total error associated with each α is determined as the average over all five folds. Doing the error assessment over five different splits reduces the dependence on the specific choice of train/test data points—this is the same rationale as used earlier regarding training over 1000 different data splits.

Once the optimal value of α has been determined, the data was fit using Lasso regression and the features are ranked based on the absolute value of the coefficients, as this orders features based on importance. Features were then added to the unregularised linear model, starting with the most important, until a minimum error was reached. The AARD, R^2, R^2_{adj} and coefficients reported for the models are the averages over all 1000 fitting iterations. This fitting approach was inspired by Beckner et al.,[149] who developed a QSPR model for the prediction of IL viscosities, using multiple train:test splits and Lasso regression for feature reduction. The code used to implement the linear and Lasso regressions, as described here, can be found in Appendices A and B, respectively.

3.5 Feed Forward Neural Network Development

The procedure used to train the FFNN is presented here (see Section 4.2 for results). First, the treatment of the train:test split is explained, after which the optimisation of network architecture and hyperparameters is described. Scikit-learn is used for preprocessing of data and hyperparameter optimisation, and the Keras Python library[150] is used to train the network.

3.5.1 Train:Test split

When training a FFNN, the weights are randomly initialised (details are given in Section 2.2.3). As a result, a FFNN can be trained multiple times with the same parameters

and input features, and produce slightly different weights, resulting in a slightly different prediction. For this reason, in this work, FFNNs were trained 30 times, each of which was used to make a prediction, and the AARD is calculated as the average over all 30 predictions. Consequently, a FFNN requires being trained 30 times more than each linear model, and to avoid this becoming too time consuming, the FFNN was trained 30 times, each time with the same single train:test split of the data, as opposed to 1000 different splits as done with the linear regression models.[V] To ensure the split was not biased towards a certain class (i.e. training the model largely on imidazolium cations, affecting performance on other classes), the data set was stratified—an equal proportion of each cation class of ILs appears in both the training and testing set. Therefore, the data set was split into a 80:10:10 train:validation:test set, stratified based on the cation class and grouped such that no particular IL occurs in more than one set. The test set was used to assess the AARD and the validation set used for hyperparameter optimisation (explained in the next section). The distribution of the stratified data set is summarised in Table 3.2, and the distribution of the full data set can be found in the ESI.

Table 3.2. Distribution of unique ILs in the training, validation, and test set, based on the cation core, used to train the FFNN.

Cation core	Training set	Validation set	Test set
Imidazolium	91	10	11
Pyrdinium	37	4	5
Ammonium	16	2	2
Phosphonium	15	2	2
Pyrrolidinium	8	1	1

V The computational expense of one training iteration of a neural network is much greater than that of linear regression, and training a network more than 30 times would become unfeasible.

3.5.2 Hyperparameter optimisation

Optimisation of a FFNN requires fine-tuning of many hyperparameters.[VI] Here, the hyperparameters were classified into two groups: (i) architectural hyperparameters and (ii) training hyperparameters. These are split up as the training hyperparameters are optimised for a specific architecture, and therefore the architecture must be determined first.

3.5.2.1 Architectural Hyperparameters

The architecture of a neural network refers to the number of layers in a network, and the number of nodes in each the input, hidden, and output layers (see Figure 4.7). The number of nodes in the input and output layers are predetermined: the input nodes are the input features and the output layer is a single node, which is the predicted value. However, the number of hidden layers and the number of nodes per hidden layer needs to be determined. The Universal Approximation Theorem claims that a FFNN with a single hidden layer containing a finite number of nodes can approximate any continuous function[151] and based on this, only FFNNs with a single hidden layer were explored here. The number of nodes in the hidden layer is determined by training a network with a single hidden node, and increasing the number of nodes until a minimum error was reached. As a network was being trained, this dids require hyperparameters to be specified, and therefore a batch size of 20, ReLU activation function, and the RMSProp optimiser were used.[VII] The hyperparameters will be discussed below in more detail.

VI Hyperparameters differ from model parameters as parameters are determined during model training (ie. weights) and hyperparameters are determined prior to training, set manually, and are used to determine model parameters.

VII These hyperparameter choices were used here, as they are typically used hyperparameters.[114]

3.5.2.2 Training Hyperparameters

Training hyperparameters were optimised for the final architecture using grid-search cross-validation, which determines a grid of hyperparameters and performs an exhaustive search, assessing all hyperparameter combinations using 5-fold cross-validation. This is implemented in Scikit-learn and is used to determine the best combination of the following hyperparameters:

1. **Activation function.** As discussed in Section 2.2.3, a nonlinear activation function is required for backpropgation. Therefore, the following activation functions were assessed:

   (a) Sigmoid

   (b) Hyperbolic tangent (tanh)

   (c) Rectified Linear Unit (ReLU)

   (d) Softmax

2. **Batch Size.** The batch size determines how many training instances (data points) are passed through the network before the weights are updated when using gradient descent. Generally, smaller batch sizes are preferred as this requires less memory.

3. **Optimiser.** Gradient descent was used to optimise weights; however, this can be slow for large data sets. Faster optimisers than regular gradient descent can be implemented and the optimisers below were assessed:

   (a) Stochastic Gradient Descent (SGD)

   (b) Adaptive Gradient Algorithm (AdaGrad)

   (c) Root Mean Square Propagation (RMSProp)

   (d) Adaptive Moment Estimation (Adam)

The following hyperparameters were left unchanged throughout training:

1. **Loss function.** The loss function is used to assess errors and the mean square error (MSE) is used here.

2. **Training epochs.** The number of epochs refers to how many times the entire data set is passed through the network, regardless of batch size. If there are too few training epochs, the data can be underfit; however, too many training epochs can lead to overfitting of the data. To minimize the risk of either under or overfitting, early stopping can be implemented. Early stopping regularises the neural network by stopping the training as soon as a minimum error is reached, instead of running through all specified training epochs. Early stopping was implemented for the final model.

The code used to implement the training of the FFNN, after architecture and hyperparameter optimisation can be found in Appendix C.

3.6 External Data Set

The linear regression and ANN fitting procedures deal with the train:test split of the data in a different manner—the linear regression reports an error over 1000 different test sets, whereas the ANN uses a single, stratified test set. As such, the AARDs produced by these models cannot be directly compared. All final models are therefore applied to the same external data set, for which the AARDs produced can then be compared. Although the training procedures ensure that the train and test sets do not overlap, some of the component ions will inevitably occur in both the train and test set. It is favourable to have a model that is generally applicable and thus to introduce a consistent test set with ions that are structurally distinct from that used during training/testing. The ILs used in such an "external set" are listed in Table 3.3, for which the structures of the cations and anions forming the species are shown in Figures 3.7 and 3.8, respectively.

All ILs included in the external set contain an ion that is not present in the training data set and applying models to these species will provide an indication of the performance when applied to "unseen" ILs. $[N_{4444}][doc]$ was included in the correlation presented by Preiss et al.;[21] however, a large error resulted for this species and it was consequently excluded from the final results reported. Including this species offers a comparison in the robustness of the models developed here to that of Preiss et al. Diedrichs and Gmehling[152] measured temperature-dependent C_P values for $[C_4apy][NTf_2]$ using multiple methods, making use of either differential scanning calorimetry or modulated-temperature differential scanning calorimetry techniques. From this study, it was determined that the Tian-Calvet calorimeter provides the most accurate measurements, and these were used here. $[C_4apy][NTf_2]$ is included in the external set due to the consequent high precision of the measurements and predicted values can be compared to experimental values without concern for inconsistent experimental results. $[C_8C_1pip][NTf_2]$ and $[C_8iquin][SCN]$ were initially present in the data set used for model development, but were removed as they were the only species found to have the respective cation cores after cleaning the data. By including these in the external set, the prediction of C_P for ILs with cation cores not used to develop the models can be assessed. C_P data for the ILs including the $[sac]^-$ anion were published after the data set was extracted from ILThermo and therefore could only be used in the external set. These ILs also provide a measure of the model performance as a result

Table 3.3. External data set used for comparison of model performance.[a]

IL	Data points	Temperature range	C_P range
$[N_{4444}][doc]$[152]	2	298–323	1325–1385
$[C_4apy][NTf_2]$[153]	48	313.13–425.15	688–739
$[C_8C_1pip][NTf_2]$[154]	12	275–385.05	720–900
$[C_8iquin][SCN]$[155]	14	278.15–343.15	510–540
$[C_4C_1im][sac]$[156]	56	293.15–348.15	557.7–597.1
$[C_6C_1im][sac]$[156]	56	293.15–348.15	629.8–671.7
$[C_8C_1im][sac]$[156]	56	293.15–348.15	694–736.1
$[C_{10}C_1im][sac]$[156]	56	293.15–348.15	763.1–815

[a] Heat capacities in kJ/mol and temperatures in K.

of varying the length of an alkyl chain of a species. Additionally, the $[sac]^-$ anion was not present in any of the ILs in the training sets.

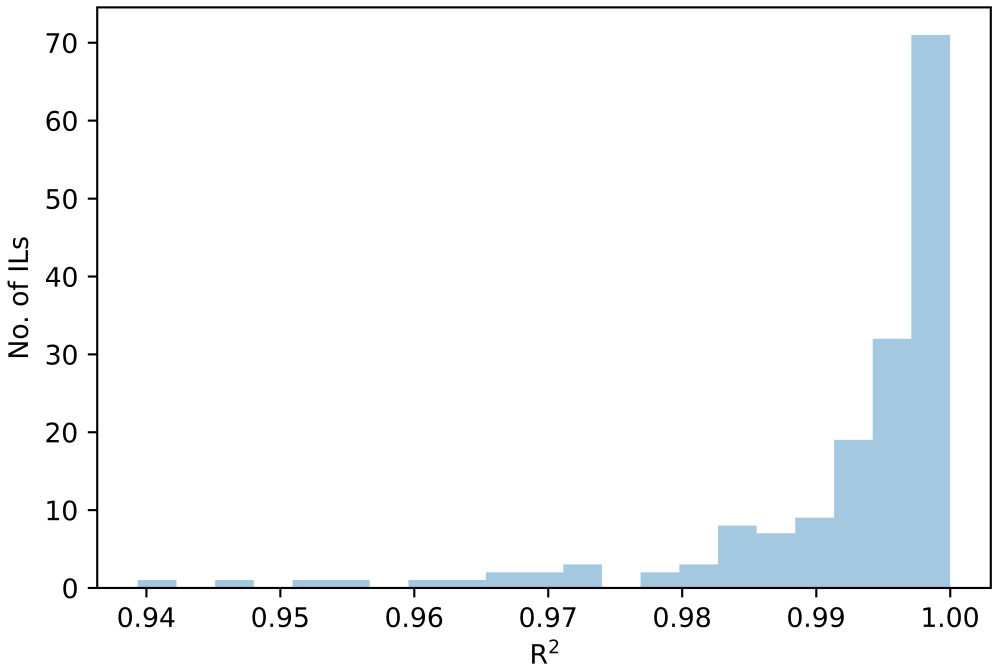


Fig. 3.5. Histogram of R^2 values for the linear correlation of C_P and temperature, assessed for all temperature-dependent species.

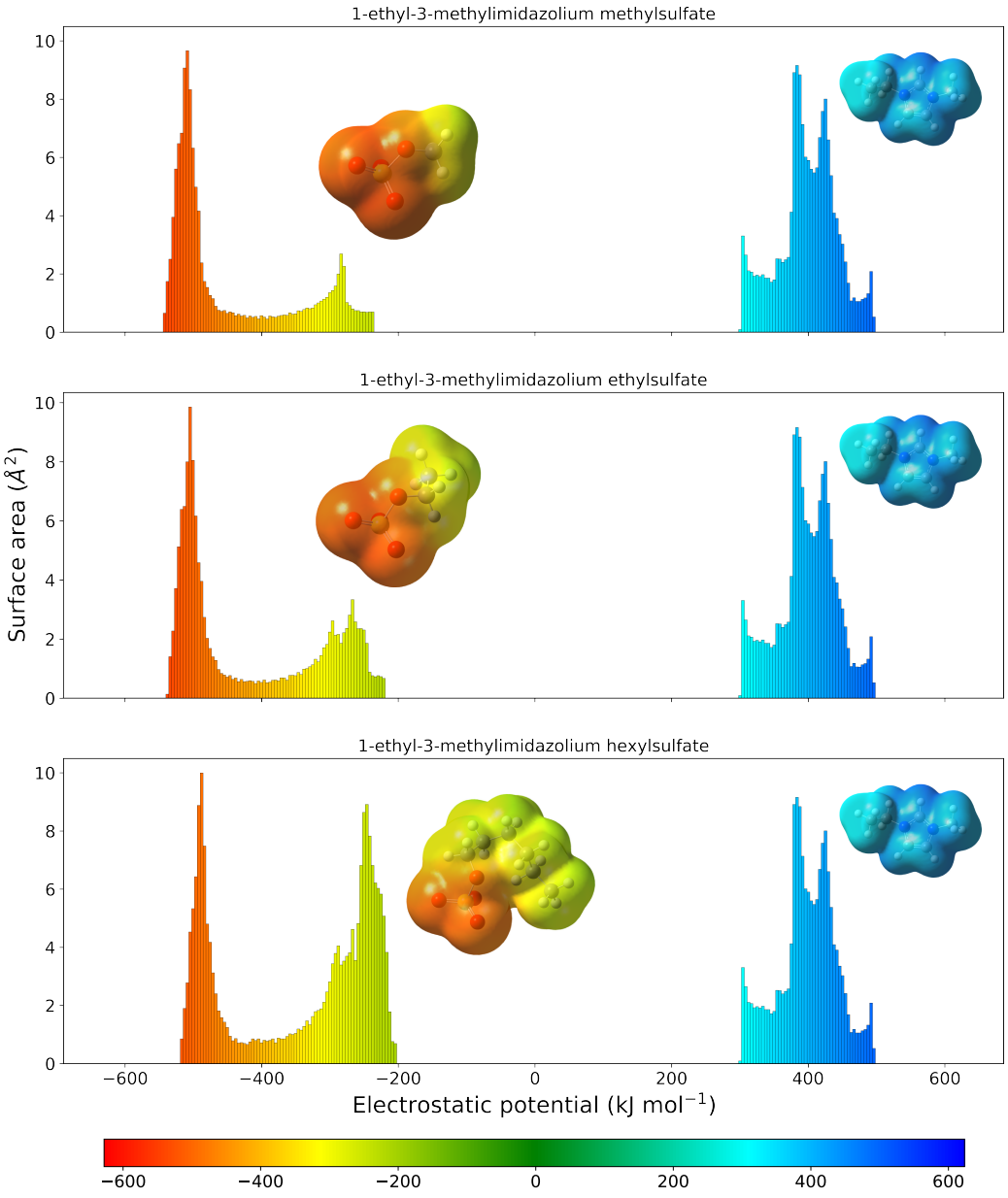


Fig. 3.6. Histograms of the electrostatic potential surface area for 1-ethyl-3-methylimidazolium alkylsulfates.

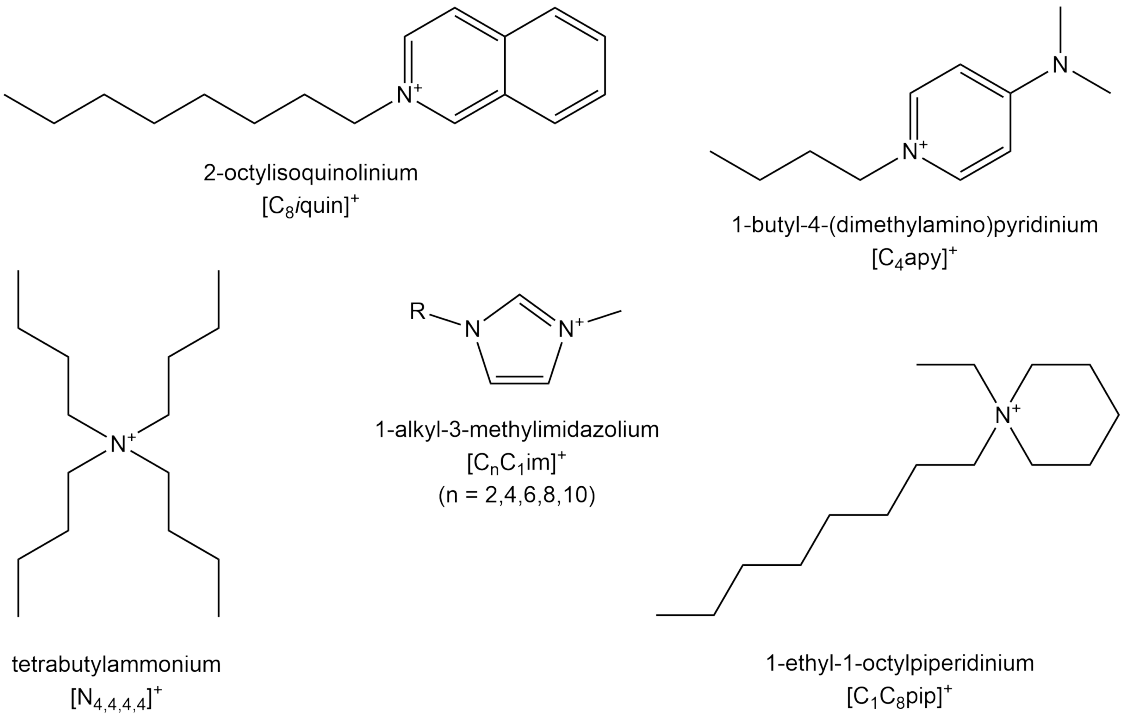


Fig. 3.7. Structures of cations used in external set.

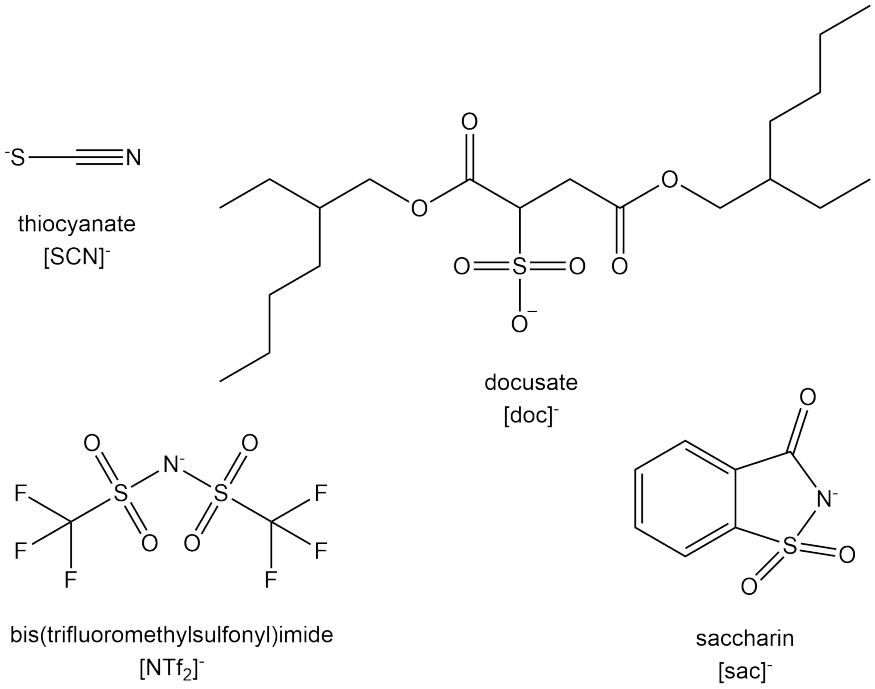


Fig. 3.8. Structures of anions used in external set.

Chapter 4

Predicting Heat Capacities

This chapter presents the training, results and discussion of the predictive models developed in this work. Models trained with linear regression are presented in the first section in the following order: (i) the baseline (BL) model; (ii) the electrostatic potential surface area (SEP) model; (iii) the GIPFs models, using individual features (GIPF-1) and product features (GIPF-2) and (iv) the QSPR model. The development of the BL model includes assessment of the methods for determining the molecular volume, V_{m}, as well as the temperature dependence of the data set as a whole. The next section details the optimisation of the hyperparameters and training of the feed-forward neural network (FFNN), developed using the GIPF features. In the final section the performance of the models when applied to the external set of ionic liquids (ILs), outlined in Section 3.6, is assessed. *It should be noted that all features used to train models were scaled.*

4.1 Linear Regression

The linear models are trained using the following procedure: A 90:10 train:test split of the data is made, ensuring that temperature-dependent points of a single IL are kept together in either the training or testing set. Where feature reduction is implemented, this is achieved using Lasso regression, optimising the regularisation parameter, α, using 5-fold

cross-validation. Regression is performed 1000 times, using a different train:test split each time, as illustrated in Figure 4.1. The final average absolute relative deviation (AARD) presented for each model is given as the average, with a standard deviation, over all 1000 regressions. Further detail on the linear regression and feature reduction procedures are given in Section 3.4.

4.1.1 Baseline Model

It was necessary to first develop a baseline (BL) model to serve as a benchmark against which model performance, as a result of different features, could be assessed. In the GCM presented by Gardas and Coutinho,[72] a linear relationship was demonstrated between C_P and V_m, and as discussed in Section 1.3.3, Preiss et al.[21] use the following simple, yet accurate correlation between V_m and C_P,

$$C_P = iV_\mathrm{m} + j. \tag{1.30}$$

It was also shown by Paulechka et al.[101] that C_P/V_m is constant at $298.15\,\mathrm{K}$, and based on these models, the linear relationship between V_m and C_P (as expressed in Equation 1.30) was chosen as the starting point for the BL model, as it only requires the determination of V_m for the data set. The original model presented by Preiss et al. was fit using C_P data at a fixed temperature, whereas the data set here consists of ILs with C_P values determined at

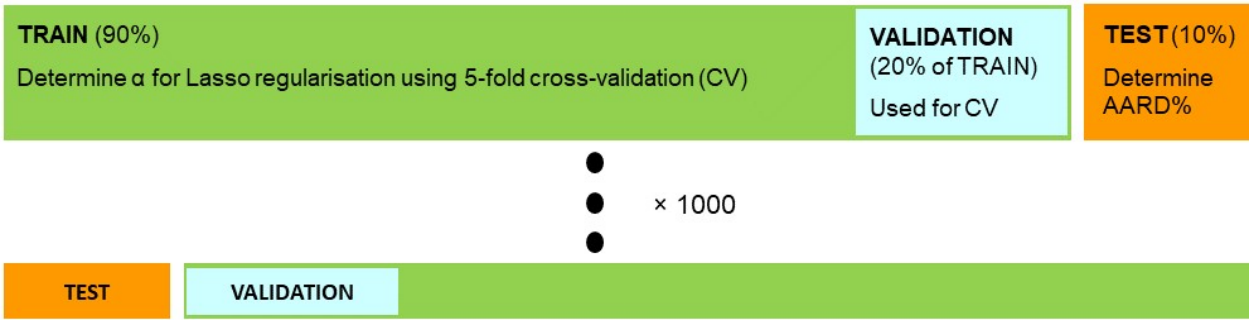


Fig. 4.1. Illustration of linear regression model development procedure used in this work.

multiple temperatures and temperature dependence must be described by the BL model. Consequently, there are two components of the BL model that were investigated: (i) the calculation of V_{m} and (ii) the temperature dependence of the data set as a whole.

4.1.1.1 Calculation of Molecular Volumes

The first step in building the BL model was determining the method that produces V_{m} values that best represent the data. The molecular volume of each ion in the data set was calculated (i) using the Solvent Excluded Surface (SES); (ii) using an electron density isosurface (ISO) and (iii) using atomic and bond contributions (ABC). V_{m} of each species was then taken as the sum of the component ion volumes, calculated using each method, resulting in (i) ${V_{\mathrm{m}}}^{\mathrm{SES}}$; (ii) ${V_{\mathrm{m}}}^{\mathrm{ISO}}$ and (iii) ${V_{\mathrm{m}}}^{\mathrm{ABC}}$. Detail on these calculation methods is given in Section 3.2.2, where the correlation between the volumes produced by each was assessed, indicating that each provide a slightly different descriptions of the data and the most accurate of these needed to be determined.

Using a subset of the data set consisting of 102 ILs, spanning the five cation families, with C_P values at a fixed temperature of $298.15\,\mathrm{K}$, the relationship between V_{m} and C_P,

$$C_P = \beta_0 + \beta_1 V_{\mathrm{m}}, \tag{4.1}$$

was modelled using Vm^{ABC}, ${V_{\mathrm{m}}}^{\mathrm{ISO}}$, and ${V_{\mathrm{m}}}^{\mathrm{SES}}$ in place of V_{m}, for which the results are given in Table 4.1. ${V_{\mathrm{m}}}^{SES}$ results in the lowest AARD; however, using ${V_{\mathrm{m}}}^{\mathrm{ABC}}$ opposed to ${V_{\mathrm{m}}}^{\mathrm{SES}}$ results in an AARD difference of only $0.21\,\%$. As ${V_{\mathrm{m}}}^{\mathrm{ABC}}$ requires a much shorter calculation time, it is a far more attractive calculation option; therefore, it is used throughout this work and further mention of V_{m} refers to ${V_{\mathrm{m}}}^{\mathrm{ABC}}$. It should be noted that the difference in AARD between the three methods is well within the standard deviation of the error, indicating that the procedure used to calculate V_{m} is unlikely to have significant impact on the quality of the prediction.

Table 4.1. Results produced by the BL model at a fixed temperature using each method for the determination of V_{m}.[a]

Volume	AARD	R^2	R^2_{adj}
V_m^{ABC}	7.51 ± 2.98	0.87 ± 0.12	0.84 ± 0.15
V_m^{ISO}	7.74 ± 3.16	0.87 ± 0.12	0.84 ± 0.15
V_m^{SES}	7.30 ± 3.04	0.86 ± 0.13	0.83 ± 0.17

[a] Heat capacities predicted at 298.15 K. Numbers are reported with standard deviations shown.

4.1.1.2 Temperature Dependence

Of the ILs in the data set with temperature-dependent points, a linear temperature dependence describes $77.44\,\%$ of the species with an $R^2 \geq 0.99$. In the work presented by Zábranský et al.,[157] C_P was determined as a function of temperature for 50 ILs using a power series, suggesting that while a linear temperature dependence describes the ILs in this data set sufficiently, including a polynomial expression for temperature could result in a more accurate BL model. Therefore, a linear, quadratic, and cubic temperature dependence were added to Equation 4.1, resulting in

$$C_P = \beta_0 + \beta_1 V_{\mathrm{m}} + \beta_2 T, \tag{4.2}$$

$$C_P = \beta_0 + \beta_1 V_{\mathrm{m}} + \beta_2 T + \beta_3 T^2 \tag{4.3}$$

and

$$C_P = \beta_0 + \beta_1 V_{\mathrm{m}} + \beta_2 T + \beta_3 T^2 + \beta_4 T^3, \tag{4.4}$$

respectively, each of which were modelled using the full data set.

Equation 4.1 was also refit using the full data set to provide a model with a "constant" temperature dependence, and the AARDs produced by these models are shown in Figure 4.2. While a polynomial expression can be used to describe the temperature dependence of an individual IL as shown by Zábranský et al., this expression differs for each species, and trying to describe the full data set with the same polynomial function will

inherently result in a higher error, due to overfitting of the data. Consequently, a simpler expression is more flexible and robust in describing the data set as a whole, and Equation 4.2 is used as the final baseline model, expressed as

$$C_P = 620.92 + 245.33V_{\mathrm{m}} + 28.19T, \tag{4.5}$$

producing an AARD of $7.28 \pm 2.08\,\%$ with an R^2 of 0.88 ± 0.11.

An interesting outcome of the comparison shown in Figure 4.2 is the notably small improvement in the accuracy of the model when temperature dependence is included. In Equation 4.5, the coefficient of V_{m} is approximately nine times larger than that of temperature, indicating that V_{m} accounts for a much larger percentage of the prediction, and temperature can be seen as a small correction. However, although small, the inclusion of temperature does result in a lower error and, as this is a temperature-dependent model, should not be excluded.

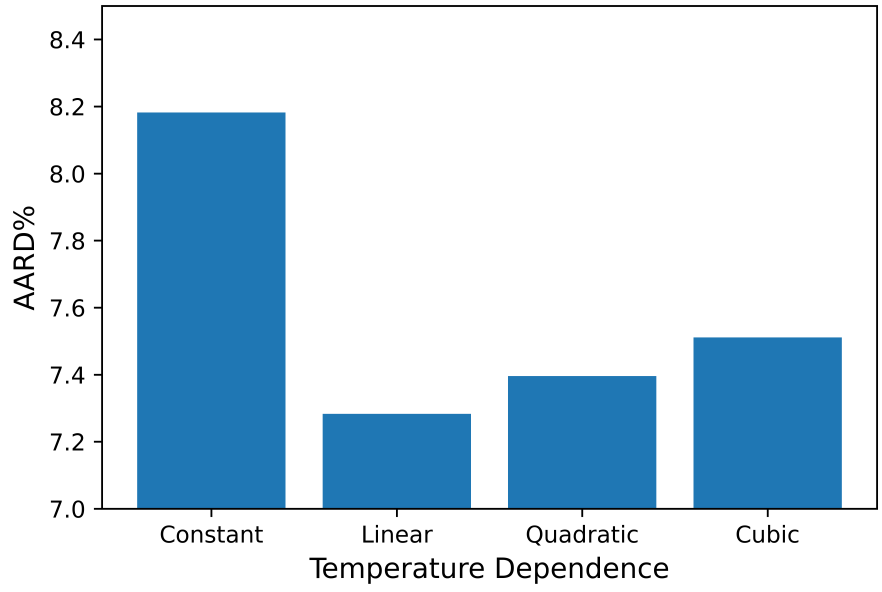


Fig. 4.2. AARD% produced from the BL model with the inclusion of a constant, linear, quadratic, and cubic temperature dependence.

4.1.1.3 Critique of Baseline Model

The BL presented here predicts C_P as a function of V_m and temperature, and only V_m is used to differentiate between different species. While the model presented by Preiss et al.[21] does indicate a sufficient correlation between V_m and C_P, it also suggests that species of the same volume will have the same C_P value at a fixed temperature, regardless of the nature of the component ions. From the data set, 1-butylpyridinium bistriflimide, [C_4py][NTf_2], and 2-methyl-1-propylpyridinium bistriflimide, [C_1C_3py][NTf_2], were identified to have the same calculated value of V_m ($306.78\,\mathrm{nm}^3$); however, at $333.15\,\mathrm{K}$, C_P values of $607\,\mathrm{J/(mol\,K)}$ and $576\,\mathrm{J/(mol\,K)}$ were reported for [C_4py][NTf_2] and [C_1C_3py][NTf_2], respectively. Therefore, although the C_P values differ by $31\,\mathrm{J/(mol\,K)}$, the BL model would predict the same value for both species. This indicates that although V_m correlates with C_P, a description of the nature of the component ions needs to be included to differentiate between species of the same V_m.

4.1.2 Electrostatic Potential Surface Areas

The electrostatic potential, ESP, provides a means by which the electrostatic interactions tendencies of a molecule can be described. Although this is a continuous property, it can be discretised and transformed to a vector of descriptors, as explained in Section 3.2.4. Based on the work of Kang et al.,[109] electrostatic potential surface area (S_{EP}) features were added to the BL model to provide a description of the electrostatics of the ions in the data set, producing the SEP model,

$$C_P = \beta_0 + \beta_1 V_\mathrm{m} + \beta_2 T + \beta_{3,i} \sum_{i=1}^{300} \beta_{3,i} S_{EP(i)}. \tag{4.6}$$

The SEP model differs from that presented by Kang et al. as V_m is used in place of molecular weights, W. To assess the effect of using V_m , rather than W, the BL model (Equation 4.2)

was also modelled using W such that

$$C_P = \beta_0 + \beta_1 W + \beta_2 T, \tag{4.7}$$

which produced an AARD of $15.60 \pm 3.06\,\%$, with an R^2 of 0.58 ± 0.32. This AARD is significantly greater than the BL models trained using V_m (see Table 4.1), and the low R^2 indicates that W does not describe the variance in the data set as well as V_m.

The SEP model presented in Equation 4.6 consists of 302 descriptors and therefore training the model is not trivial. As discussed in Section 2.2.2, a linear model with too few descriptors will underfit the data; however, too many descriptors will overfit the training data, producing large errors when applied to a test set. Finding the ideal number of features is achieved through regularisation of the model. Lasso regression (as described in Section 3.4.2) was used to reduce the feature space and Figure 4.3 shows that the lowest AARD results when the top four features, ranked by the absolute value of the corresponding fitting coefficient, are used. These selected features resulted in the final SEP model,

$$C_P = \beta_0 + \beta_1 V_\mathrm{m} + \beta_2 T + \beta_3 S_{EP(248.95)} + \beta_4 S_{EP(-274.05)}, \tag{4.8}$$

for which the values of β_0 to β_4 are given in Table 4.2, producing an AARD of $7.18 \pm 2.07\,\%$ with an R^2 of 0.88 ± 0.10. It should be noted that the value i of each $S_{EP(i)}$ is the midpoint of the histogram bins described in Section 3.2.4.1; for example, $S_{EP(248.95)}$ does not refer to the area with an ESP of exactly $248.95\,\mathrm{kJ/(mol\,K)}$, but rather the surface area with an ESP between $246.86\,\mathrm{kJ/(mol\,K)}$ and $251.04\,\mathrm{kJ/(mol\,K)}$.

4.1.2.1 Discussion

The BL model presented in Section 4.1.1 produced an AARD of $7.28 \pm 2.08\,\%$ and the SEP model produced an AARD% of $7.18 \pm 2.07\,\%$. Since the ARRD is an average over a number of different test sets, it is unfortunately not possible to extract the root cause of the

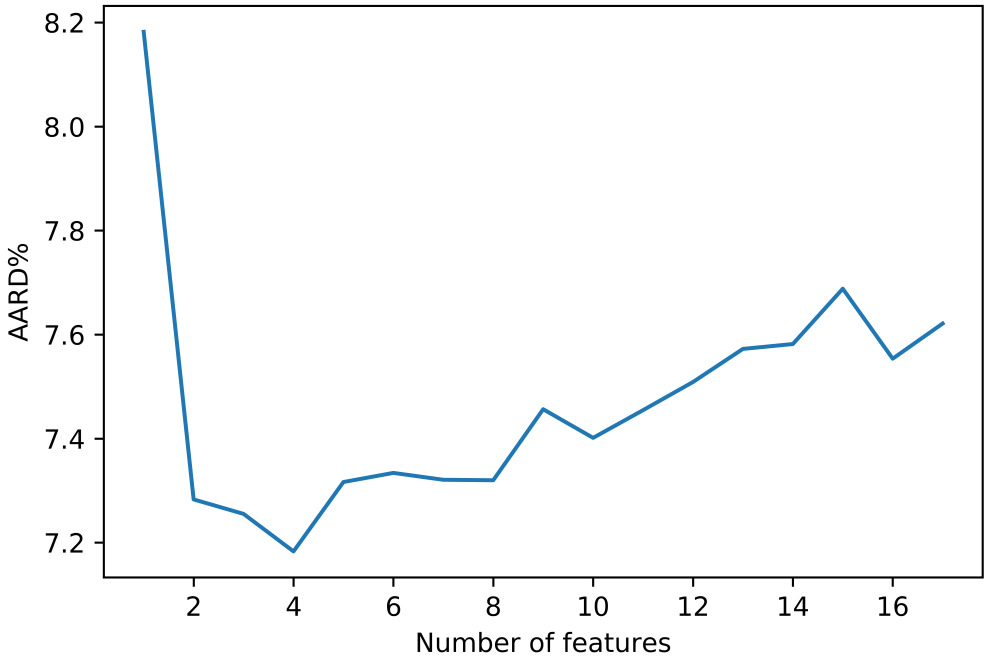


Fig. 4.3. AARDs produced with sequential adding of features to the SEP model, indicating minimum error when four features are used.

improvement; however, albeit small, the inclusion of the S_{EP} features improves on the BL model. The feature reduction procedure reduced the initial 302 features to V_{m}, T, $S_{EP(248.95)}$, and $S_{EP(-274.05)}$, of which V_{m} makes the largest contribution, describing the bulk of the prediction. T, $S_{EP(248.95)}$, and $S_{EP(-274.05)}$ all make similar contributions, and together form approximately one third of the absolute value of the prediction. The two S_{EP} features that are used in the final model can be interpreted as a characterisation of the polarity of the cation and anion; since the $S_{EP(248.95)}$ feature is the area corresponding to a positive ESP, which would only be found on cations, and the $S_{EP(-274.05)}$ feature corresponds to a negative ESP, only found on the anions. Therefore, these allow for discrimination between ILs that is not solely based on the ion-pair molecular volume, but also on the electrostatic character of the cation and anion.

A flaw with the S_{EP} features is the inability to describe all ions, that being, species that do not have any regions with an ESP in the range of $246.86\,\mathrm{kJ/(mol\,K)}$ to $251.04\,\mathrm{kJ/(mol\,K)}$

Table 4.2. Average coefficient values of the final SEP model.[a]

Coefficient	Value
β_0	620.93 ± 8.76
β_1	206.65 ± 12.51
β_2	29.00 ± 2.75
β_3	31.07 ± 3.68
β_4	28.40 ± 4.28

[a] Coefficients are reported with standard deviations shown.

or $-244.96\,\mathrm{kJ/(mol\,K)}$ to $-249.14\,\mathrm{kJ/(mol\,K)}$, are not described by the S_{EP} features used in the model. Of the 208 ILs in the data set, 16 have neither of these features and are essentially being described by the BL model, therefore this model makes no improvement on the description of these species. For a greater improvement, it is necessary to add features that better describe the variability of the data set.

4.1.3 General Interaction Properties Functions

The GIPF features described in Section 3.2.4.2 offer an alternative featurization of the ESP. As these are based on statistical measures of the variation in the ESP on the surface of a molecule, each GIPF feature can be determined for every IL in the data set and zero-valued features will only result if the variance is zero. The GIPF features were added to the baseline model in place of the S_{EP} features resulting in

$$C_P = \beta_0 + \beta_1 V_\mathrm{m} + \beta_2 T + \beta_3 \sigma_+^2 + \beta_4 \sigma_-^2 + \beta_5 \Pi_+ + \beta_6 \Pi_-, \tag{4.9}$$

referred to as the GIPF-1 model. As this model consists of only six features, feature reduction was not necessary and thus multiple linear regression was used, without regularisation, according to the procedure outlined in Section 3.4. This resulted in an AARD of $7.35 \pm 2.02\,\%$ with an R^2 value of 0.88 ± 0.11. The AARD produced by this model is not only worse than the of the SEP model, but also higher than that of the BL model and the

relatively low R^2 value indicates that the GIPF-1 model does a worse job of describing the variance than the previous two models. A possible reason for this decrease in quality is discussed in the next section.

4.1.3.1 Critique of Individual Contributions

All models developed up to this point consist of features generated for the cation and anion separately, based on the assumption that the ions will make a fixed contribution to C_P, regardless of the strength and mode of interaction between the ions. Such additivity is used by Gardas and Coutinho[72], and Müller and Albert[76] present an example to show that additivity of ions is sound.

Considering two cations, $[X_1]^+$ and $[X_2]^+$, and two anions, $[A_1]^-$ and $[A_2]^-$, if each anion makes a fixed contribution to C_P, then

$$C_P([X_1][A_1]) - C_P([X_1][A_2]) = C_P([X_2][A_1]) - C_P([X_2][A_2]) \tag{4.10}$$

should hold true. The above relationship was investigated for a subset of the data and the selected results are presented in Table 4.3. The similar differences produced in which structurally similar $[CnC_1im]^+$ cations (a) are compared would suggest that the anion contributions are close to constant. However, when comparing different cation cores with the same alkyl chains (b), and different cation cores with different alkyl chains (c), the anion contributions are not fixed and show marked deviation. Additivity of individual ion heat capacities is fundamental to the principles behind GCM and VBT models; however, the information presented here indicates that, although these models provide meaningful correlations, the interactions between ions need to be described to provide a more robust description of C_P. The shortcomings of using ion additivity was addressed by He et al.[88] in which it is pointed out that the ion-ion interaction needs to be considered and the product of cation and anion features were included to account for these interactions.

Table 4.3. Differences in the anion contribution to C_P across: (a) cations differing in alkyl chain length; (b) different cation cores with the same alkyl chains and (c) different cation cores with different alkyl chains, indicating the breakdown of the additivity assumption made in Equation 4.10.[a]

	C_P	
a.	$[C_4C_1im]^+$	$[C_2C_1im]^+$
$[OAc]^-$	381(23)	321.9(9.7)
$[BF_4]^-$	366.7(1.9)	304.5(6.5)
Diff.	−14.3	−17.4
b.	$[C_2C_2im]^+$	$[C_2C_2py]^+$
$[C_2SO_4]^-$	413(16)	412(7)
$[NTf_2]^-$	533(12)	566(13)
Diff.	120	154
c.	$[C_4C_1py]^+$	$[C_6C_1im]^+$
$N(CN)_2]^-$	368(22)	534(54)
$[BF_4]^-$	412.3(6.4)	427.8(6.6)
Diff.	44.3	−106.2

[a] The cations used are 1-butyl-3-methylimidazolium ($[C_4C_1im]^+$), 1-ethyl-3-methylimidazolium ($[C_2C_1im]^+$),1,3-diethylimidazolium ($[C_2C_2im]^+$), 1,3-diethylpyridinium ($[C_2C_2py]^+$), 1-butyl-2-methylpyridinium ($[C_4C_1py]^+$) and 1-hexyl-3-methylimidazolium ($[C_6C_1im]^+$). The anions used are acetate ($[OAc]^-$), tetrafluoroborate ($[BF_4]^-$), ethylsulfate ($[C_2SO_4]^-$), bistriflimide ($[NTf_2]^-$), and dicyanamide ($N(CN)_2]^-$). All heat capacities and differences are in J/(mol K) at 298 K, with the experimental error reported on ILThermo indicated in parentheses.

4.1.3.2 Interacting Features

To create interacting features, products to the third order were made of the six features used in the GIPF-1 model (Equation 4.9). This resulted in 83 input features, which were reduced in the same manner as the S_{EP} features using Lasso regression, as outlined in Section 3.4.2. As shown in Figure 4.4, the lowest AARD resulted when the model consisted

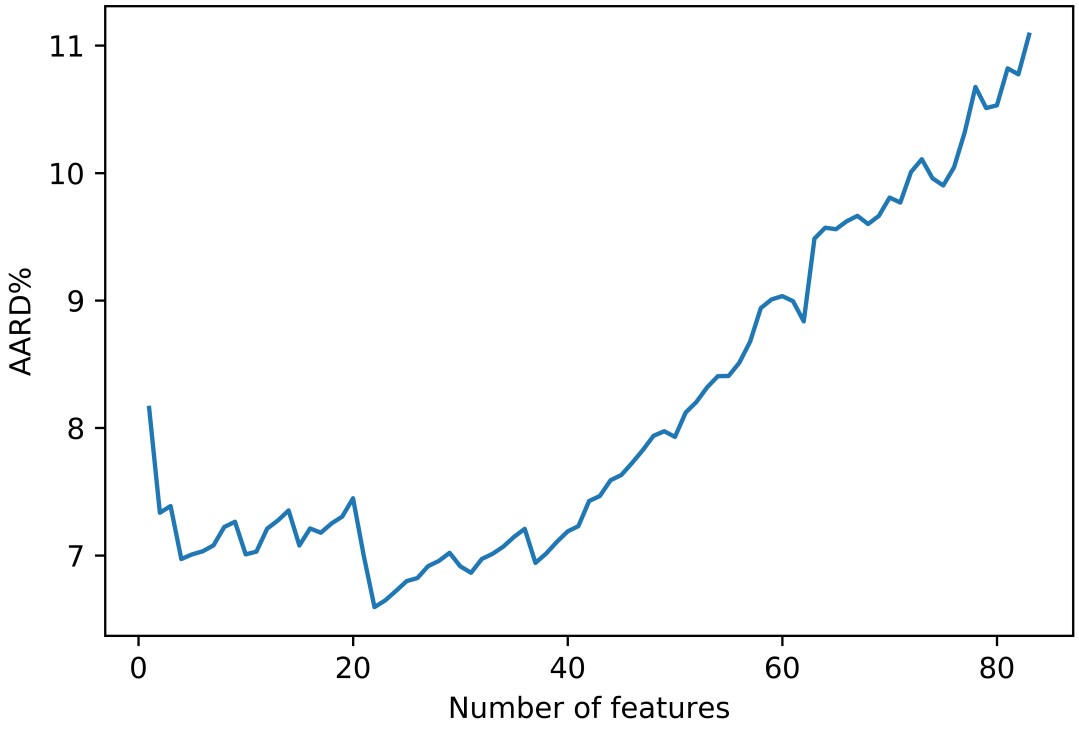


Fig. 4.4. AARDs produced with sequential adding of features to the GIPF-2 model, indicating a minimum error when 23 features are used.

of the top 23 features, resulting in the GIPF-2 model,

$$\begin{aligned} C_P =& \beta_0 + \beta_1 V_{\mathrm{m}} \sigma_-^2 - \beta_2 \sigma_+^2 \sigma_-^2 \Pi_+ - \beta_3 (\sigma_-^2)^2 + \beta_4 \sigma_-^2 \Pi_-^2 - \beta_5 V_{\mathrm{m}} \Pi_-^2 - \beta_6 {V_{\mathrm{m}}}^2 \sigma_-^2 \\ &+ \beta_7 (\sigma_+^2)^2 \sigma_-^2 + \beta_8 {V_{\mathrm{m}}}^2 \Pi_+ + \beta_9 \sigma_-^2 \Pi_+ \Pi_- - \beta_{10} \Pi_-^2 + \beta_{11} V_{\mathrm{m}} \Pi_- \\ &- \beta_{12} (\sigma_-^2)^2 \Pi_+ - \beta_{13} {V_{\mathrm{m}}}^2 \sigma_+^2 - \beta_{14} V_{\mathrm{m}} \sigma_+^2 + \beta_{15} V_{\mathrm{m}} \sigma_+^2 \sigma_-^2 + \beta_{16} T \Pi_+^2 + \beta_{17} V_{\mathrm{m}} T \\ &+ \beta_{18} V_{\mathrm{m}} (\sigma_-^2)^2 + \beta_{19} \sigma_+^2 \Pi_-^2 + \beta_{20} V_{\mathrm{m}} T \Pi_- - \beta_{21} V_{\mathrm{m}} T \sigma_+^2 + \beta_{22} V_{\mathrm{m}} - \beta_{23} T \Pi_+. \end{aligned} \tag{4.11}$$

The GIPF-2 model produced an AARD of $6.65 \pm 1.94\,\%$ and an R^2 of 0.91 ± 0.10, for which the coefficient values are given in Table 4.4 and the relative sizes are illustrated in Figure 4.5.

Table 4.4. Average coefficient values of the GIPF-2 model.[a]

Coefficient	Average Value
β_0	620.83 ± 8.76
β_1	832.20 ± 110.23
β_2	-669.65 ± 91.91
β_3	-588.06 ± 80.48
β_4	576.95 ± 68.03
β_5	-536.59 ± 71.32
β_6	-462.81 ± 75.86
β_7	459.81 ± 62.55
β_8	379.92 ± 53.26
β_9	373.63 ± 104.53
β_{10}	-243.26 ± 20.14
β_{11}	194.12 ± 31.16
β_{12}	-185.80 ± 80.07
β_{13}	-164.98 ± 45.55
β_{14}	-106.39 ± 31.36
β_{15}	91.21 ± 53.03
β_{16}	78.77 ± 16.67
β_{17}	67.37 ± 32.61
β_{18}	61.26 ± 35.81
β_{19}	49.16 ± 33.03
β_{20}	37.34 ± 14.15
β_{21}	-32.24 ± 34.90
β_{22}	27.36 ± 28.31
β_{23}	-7.97 ± 10.93

[a] Coefficients are reported with standard deviations shown.

4.1.3.3 Model Interpretation

Note that the use of products of T, $V_{\mathrm{m}}$and the GIPF features results in a significant decrease in the AARD, compared to all other linear regression models developed in this work. When products are made of the features, both univariate and multivariate (product) features are present; however, through Lasso regularisation, all univariate features, bar V_{m}, are removed from the final model, indicating the importance of combination features. A notable characteristic of the GIPF-2 model is the presence of V_{m}, which occurs in 13 out of the 23 features, again drawing attention to the importance of V_{m} in describing C_P. A second key outcome of this model is the treatment of the temperature dependence. In the BL model,

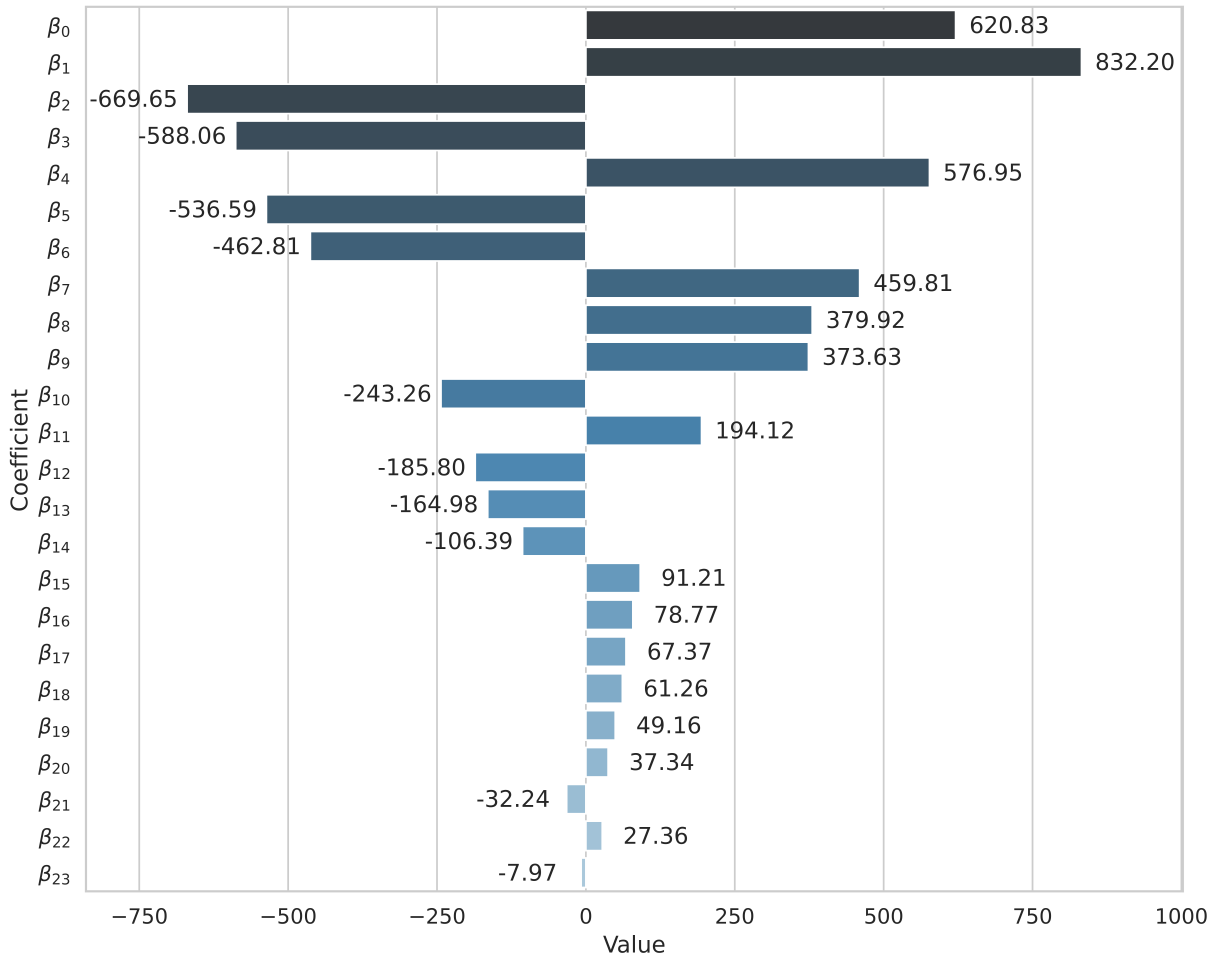


Fig. 4.5. Average coefficient values of the GIPF-2 model.

the coefficient of temperature is independent of the IL; hence, all ILs have the same linear temperature dependence. In this model however, temperature is present in five of the product features and, through combining temperature with other features unique to each IL, the temperature dependence of each IL is described differently. The last factor that is important to point out is the contribution of ion-ion interaction to the heat capacity. In the GIPF-2 model, six features are combinations of anion and cation features, providing ion-interaction terms, lacking in previous models, which could account for the significant improvement in the accuracy of this model.

4.1.4 QSPR Model

QSPR models have been used with success in literature to predict IL heat capacities.[82–86,88] The development of a QSPR model, while not a main objective of the project, provides a means for comparison between different types of linear predictive models. This section presents the results of a basic linear QSPR model using descriptors obtained from the Mordred software.[140] A total of 1825 0D to 3D descriptors[81] were obtained for the cation and anion separately, resulting in a total of 3650 features for each IL. Certain features could be trivially removed as they returned zero values for the species concerned. For example, one of the features that is calculated is the number of iodine atoms present, and since none of the ILs in this data set contain iodine, this feature will return a value of zero for every species. Therefore, features which return the same value for every ionic liquid are removed. In addition, it is often the case when faced with such a large number of features that some of these may correlate with each other. If two features are highly correlated, there is little purpose in using both, as they provide the same description of the variance in the data. A correlation matrix was constructed to identify any highly correlated features, and if two features had a correlation coefficient greater than 0.95, one of these was dropped. After removing non-descriptive and highly correlated features, 582 remained: 360 and 221 features for the cation and anion, respectively, along with temperature.

The feature space was further reduced using Lasso regularisation as described in Section 3.4.2, and it was found that the lowest AARD resulted when the top 13 features were used (see Figure 4.6). It should be noted that the AARD remained low up to 21 features; however, the least number of features was selected as this provides the simplest model. This resulted in the final QSPR model expressed as

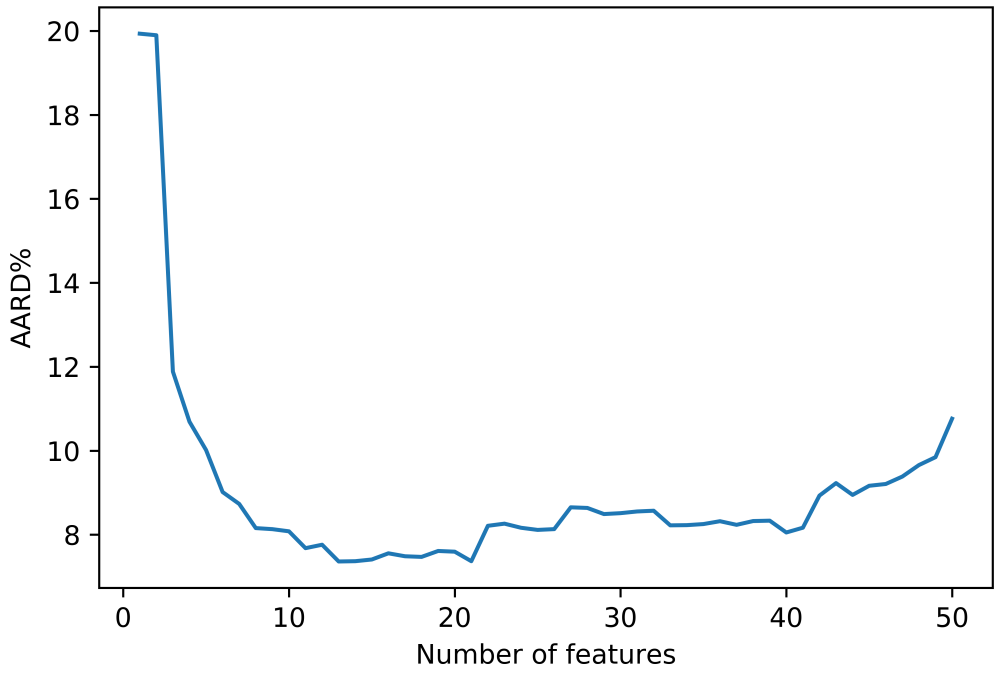


Fig. 4.6. AARDs produced with sequential adding of features to the QSPR model, indicating minimum error when 13 features are used.

$$\begin{aligned} C_P = &\beta_0 + \beta_1 \mathrm{ABC}_+ + \beta_2 \mathrm{RotRatio}_+ + \beta_3 \mathrm{VR1\_A}_- + \beta_4 \mathrm{T} + \beta_5 \mathrm{IC2}_+ + \beta_6 \mathrm{LogEE\_A}_- \\ &+ \beta_7 \mathrm{AATSC3c}_+ + \beta_8 \mathrm{AATSC1pe}_+ + \beta_9 \mathrm{JGI3}_- + \beta_{10} \mathrm{Xp\text{-}6dv}_- + \beta_{11} \mathrm{ATSC7d}_+ \\ &+ \beta_{12} \mathrm{nCl}_- + \beta_{13} \mathrm{TIC1}_-, \end{aligned} \tag{4.12}$$

where the subscripts $+$ and $-$ indicate that the feature was calculated for the cations or anions, respectively. The coefficient values for the model are given in Table 4.5 and feature names are given in Table 4.6. This model produced an AARD of $7.36 \pm 1.77\,\%$, which is the largest error so far, albeit marginally with respect to the initial GIPF-1 model (Equation 4.9).

4.1.4.1 Molecular Descriptors

Presented here are the mathematical procedure used to determine each of the molecular descriptors in Table 4.6. The definitions of many molecular descriptors are based on, or

Table 4.5. Average coefficient values of the final QSPR model.[a]

Coefficient	Average Value
β_0	620.92 ± 8.76
β_1	109.09 ± 7.15
β_2	44.68 ± 4.05
β_3	43.56 ± 6.319
β_4	34.20 ± 2.80
β_5	-33.81 ± 2.58
β_6	32.11 ± 2.87
β_7	-23.64 ± 1.87
β_8	-21.37 ± 1.98
β_9	21.19 ± 2.63
β_{10}	18.67 ± 6.82
β_{11}	-17.76 ± 3.19
β_{12}	-15.96 ± 5.02
β_{13}	15.07 ± 4.37

[a] Numbers are reported with standard deviations shown.

require an understanding, of graph theory; but, have been defined here in terms of molecular structure, based on the definitions presented in the *Handbook of Molecular Descriptors* by Todeschini and Consonni.[81] Where the definition is taken from another source, the reference is given.

ABC The *atom bond connectivity (ABC)* is calculated using

$$\mathrm{ABC} = \sum_{i,j \in bonds} \sqrt{\frac{d_i + d_j - 2}{d_i d_j}}, \tag{4.13}$$

where i and j are atoms that share a bond and d_i and d_j are the number of first neighbours of (atoms connected to) i and j, respectively.[158]

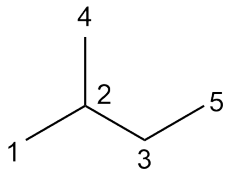
RotRatio The *rotatable bonds ratio*,[159] is the ratio of the number of rotatable bonds, $N_{\text{rotatable bonds}}$, to the total number of bonds, $N_{\text{total bonds}}$, and is expressed as

$$\mathrm{RotRatio} = \frac{N_{\text{rotatable bonds}}}{N_{\text{total bonds}}}. \tag{4.14}$$

Table 4.6. Description of the features used in the final QSPR model.

Feature	Description
ABC	Atom-bond connectivity index[158]
RotRatio	Rotatable bonds ratio[159]
VR1_A	VR1 of adjacency matrix[81,159]
T	Temperature
IC2	2-ordered neighborhood information content[140]
LogEE_A	LogEE of adjacency matrix[159]
AATSC3c	Averaged and centered Moreau-Broto autocorrelation of lag 3_{140} weighted by Gasteiger charge
AATSC1pe	Averaged and centered Moreau-Broto autocorrelation of lag 1_{140} weighted by Pauling EN
JGI3	3-ordered mean topological charge[140]
Xp-6dv	6-ordered Chi-path weighted by valence electrons[140]
ATSC7d	Centered Moreau-Broto autocorrelation of lag 7_{140} weighted by sigma electrons
nCl	Number of chlorine atoms[159]
TIC1	1-ordered neighborhood total information content[140]

VR1_A The *adjacency matrix*, $\boldsymbol{A}$, is used to indicate the connectivity within a molecule, having the dimensions $A \times A$, where A is the number of atoms in the molecule. Each element of the matrix, a_{ij}, has a value of 1 if atoms i and j are adjacent, and a value of 0 if not.[159] This is illustrated for 2-methylbutane below:



$$\boldsymbol{A} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}.$$

VR1 is a matrix aggregation method implemented in Mordred and calculated for $\boldsymbol{A}$ as

$$\mathrm{VR1}(\boldsymbol{A}) = \sum_{i,j \in bonds} (l_i \cdot l_j)^{-\frac{1}{2}}, \tag{4.15}$$

where l_i and l_j are the eigenvectors corresponding to the leading eigenvalues of rows i and j where i and j are adjacent (bonded) atoms. Using this, the VR1 of $\boldsymbol{A}$, VR1_A, is obtained.

LogEE_A LogEE is another matrix aggregation method applied to the adjacency matrix A (as defined in 4.1.4.1) and is calculated using

$$\mathrm{LogEE}(\boldsymbol{A}) = \log\left(\sum_{n=1}^{N} e^{(\lambda_n)}\right), \tag{4.16}$$

where λ_n is the eigenvector of the n^{th} eigenvalue of $\boldsymbol{A}$.[159]

IC2 The *neighbourhood information content*, IC_r, is defined as

$$IC_r = -\sum_{g=1}^{G} p_g \cdot \log_2 p_g, \tag{4.17}$$

in which p_g is the probability of finding an atom of the g^{th} element type on the r^{th} neighbour. Here, $r = 2$ and therefore, p_g is the probability of finding a certain element on the second neighbour of each atom in the molecule.

AATSC3c The *Moreau-Broto autocorrelation (ATS)* is defined as

$$\mathrm{ATS}_k = \frac{1}{2}\left(\boldsymbol{w}^{\boldsymbol{T}} \cdot^{k} \boldsymbol{B} \cdot \boldsymbol{w}\right), \tag{4.18}$$

where k is the lag—a defined path distance between two atoms and $\boldsymbol{w}$ is an A-dimensional vector of an atomic property. ${}^{k}\boldsymbol{B}$ is determined by comparing the distance between atoms i and j, d_{ij}, to k, such that

$${}^{k}B_{ij} = \begin{cases} 1, & \text{if } d_{ij} = k \\ 0, & \text{if } d_{ij} \neq k \end{cases}. \tag{4.19}$$

From this, the averaged Moreau-Broto autocorrelation (AATS) is calculated as

$$\mathrm{AATS}_k = \frac{\mathrm{ATS}_k}{\Delta_k}, \tag{4.20}$$

where Δ_k is the number of d_{ij} values that are equal to k. The AATS$_3$ is calculated, where the atomic property used is the Gasteiger charge,[160] producing the AATSC3c feature.

AATSC1pe This feature is the AATS of lag 1, where the atomic property used is the Pauling electronegativity.

JGI3 This feature is the mean topological charge index, which is obtained by first calculating $\boldsymbol{M}$, expressed as

$$\boldsymbol{M} = \boldsymbol{A} \cdot \boldsymbol{D^{-2}}, \tag{4.21}$$

where $\boldsymbol{A}$ is the adjacency matrix, and $\boldsymbol{D^{-2}}$ is the reciprocal square distance matrix determined as

$$D_{ij}^{-2} = \begin{cases} \frac{1}{d_{ij}^2}, & \text{if } i \neq j \\ 0, & \text{if } i = j \end{cases}. \tag{4.22}$$

The elements of the charge term matrix, $\boldsymbol{CT}$, are then determined from the elements of $\boldsymbol{M}$, M_{ij}, such that

$$CT_{ij} = \begin{cases} \delta_i, & \text{if } i = j \\ M_{ij} - M_{ji}, & \text{if } i \neq j \end{cases}, \tag{4.23}$$

where δ_i is the number of atoms connected to i. Using this, the *topological charge index*, G_k, can be determined as

$$G_k = \frac{1}{2} \sum_{i=1}^{A} \sum_{j=1}^{A} |\boldsymbol{CT}_{ij}| \cdot \delta(k, d_{ij}), \tag{4.24}$$

in which $\delta(k, d_{ij})$ is the Kronecker delta, such that

$$\delta(k, d_{ij}) = \begin{cases} 1, & \text{if } d_{ij} = k \\ 0, & \text{if } d_{ij} \neq k \end{cases}, \tag{4.25}$$

where k is the number of atoms between i and j. Using this, the mean topological charge index, J_k, can be calculated as

$$J_k = \frac{G_k}{A-1}, \tag{4.26}$$

where A is the number of atoms in the molecule. J_3 is denoted as JGI3 in the Mordred software.

Xp-6dv Xp-6dv is the so-called "χ-index" of order 6, which is defined using the 6-atom "paths" in a molecule. A path is a sequence of m connected atoms, and two paths in the same molecule can contain the same atom. The χ-index of order m is defined as

$$^{m}\chi = \sum (^{m}C), \tag{4.27}$$

where

$$^{m}C = \prod \left(\delta_i^V\right), \tag{4.28}$$

in which δ_i^V is determined for each atom i in a path as

$$\delta_i^V = Z_i^V - h_i, \tag{4.29}$$

where Z_i^V is the number of valence electrons on atom i and h_i is the number of hydrogen atoms connect to atom i.[161] This feature, Xp-6dv uses a path of six atoms, and therefore $m = 6$, and ^{6}C is the product of all σ_i in all six atom paths in a molecule. Once all ^{6}C values are calculated, $^{6}\chi$ can be determined to obtain Xp-6dv.

ATSC7d ATSC7de is the Moreau-Broto autocorrelation using $k = 0$, and the feature used is the number of sigma electrons on atom i.

nCl nCl is the number of chlorine atoms present in the ion.

TIC1 The neighbourhood total information content, TIC_r, is calculated as

$$\mathrm{TIC}_r = A \times \mathrm{IC}_r, \tag{4.30}$$

where IC_r is the neighbourhood information content as defined in 4.1.4.1 and A is the total number of atoms. Here, TIC_r with $r = 1$ is calculated, giving the TIC1 feature.

4.1.4.2 Feature Interpretation

The final model was comprised of 12 molecular descriptors, as well as temperature. The inclusion of temperature is necessary to discriminate between temperature-dependent points of a species, and its presence after feature reduction, as with previous models, is a testament to the ability of the automated regularisation procedure to identity important/significant variables. Of the 12 molecular descriptors, six are cation descriptors, and the remaining six are anion descriptors. The equal number of features for each ion indicates that an equal amount of information is drawn from both. An interesting outcome of this model is that V_m is dropped; whereas in all previous linear models V_m has been the most prominent feature, yet it is not present in this final model. V_m is calculated here using the same method as that used in the previous models, and its absence indicates that although good at describing C_P, other features can provide a similar description. An issue with these features is the lack of interpretability—it is not trivial to relate any of these molecular descriptors to the thermodynamic factors that influence the heat capacity of ILs.

4.2 Feed-Forward Neural Network

The linear models developed in this work suggest that the GIPF features presented in Section 3.2.4.2 are the most effective in describing C_P and based on this, were used to develop a FFNN using the following procedure: The architecture was determined first,

by setting all hyperparameters to typical values[114] and increasing the number of nodes in a single hidden layer from 1 to 20, noting which gave the lowest AARD. Once the architecture had been optimised, the optimal batch size, optimiser, and activation function were determined using 5-fold grid-search cross-validation. Due to the random initialisation of the weights, as explained in Section 2.2.3, the network was trained 30 times, using a single stratified train:test set, and an average model is reported. Further details on the FFNN training procedure are given in Section 3.5.1.

4.2.1 Architecture

The optimal architecture was determined to be a single hidden layer containing seven nodes, illustrated in Figure 4.7. It should be noted that the AARD is significantly higher when fewer nodes are used, yet when more than seven nodes are used, the error does not increase notably, and similar results can be expected (see Figure 4.8). However, seven nodes were chosen, as it is favourable to use a lesser number of nodes to reduce the complexity of the network, and therefore the time required for training.

4.2.2 FFNN Optimisation

All hyperparameters were optimised using the procedure outlined in Section 3.5.2.2 and are summarised in Table 4.7. From this, it was determined that the optimal hyperparameters are a batch size of 10, the Adam optimiser and the ReLu activation function. The final FFNN was trained as described in Section 3.5.1 using the optimal architecture and hyperparameters, and produced an AARD of $2.48 \pm 3.49\,\%$. Due to the complex connectivity of the input features, interpretability of the feature importance, as done with the linear models, is not possible.

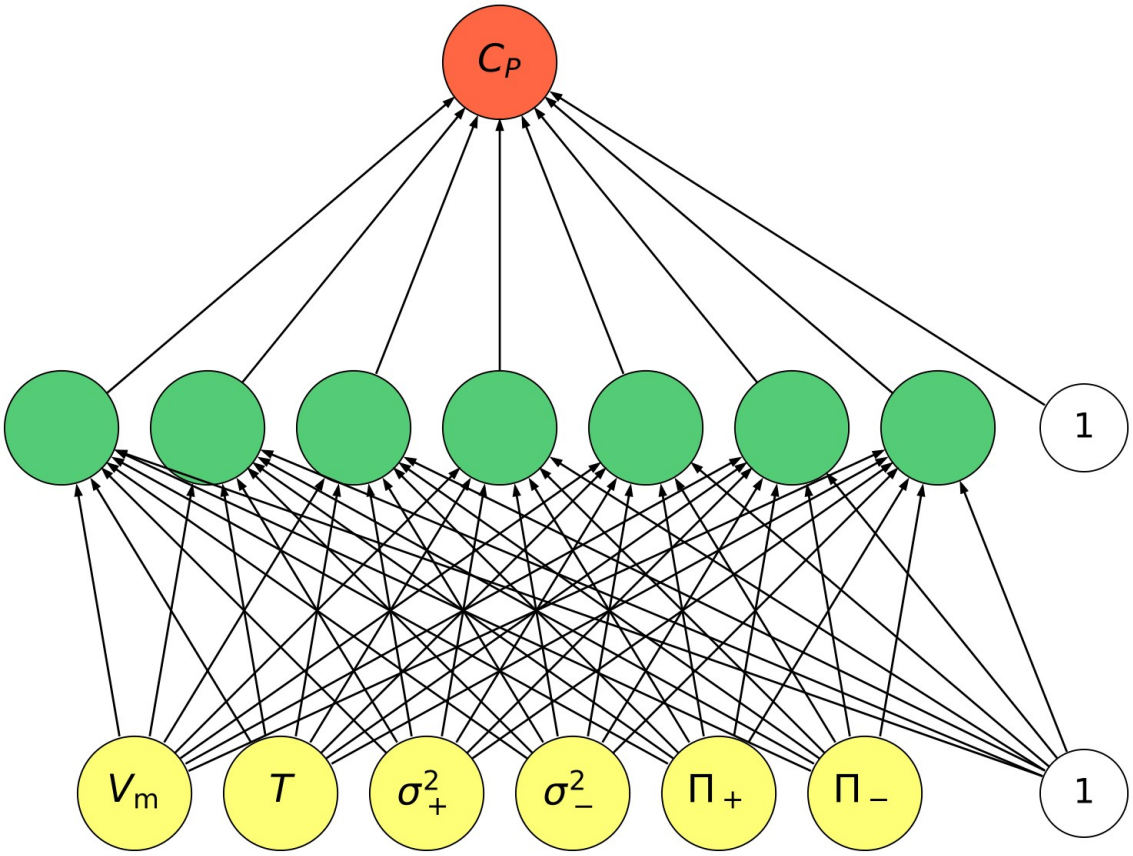


Fig. 4.7. Optimal architecture determined for the FFNN.

4.3 Summary of Results

Presented in Table 4.8 are the results of all models developed in this work.

Table 4.8. Summary of models developed in this work.[a]

Model	Feature reduction	Regression	AARD%	R^2_{adj}
BL	None	Linear	7.28 ± 2.08	0.88 ± 0.11
SEP	Lasso[b]	Linear	7.18 ± 2.07	0.88 ± 0.10
GIPF-1	None	Linear	7.35 ± 2.02	0.88 ± 0.12
GIPF-2	Lasso[c]	Linear	6.35 ± 1.94	0.90 ± 0.11
QSPR	Lasso[d]	Linear	7.36 ± 1.77	0.91 ± 0.08
FFNN	None	FFNN	2.48 ± 3.49	0.98 ± 0.07

[a] Numbers are reported with standard deviations shown.
[b] 302 features reduced to 4 features.
[c] 84 features reduced to 23 features.
[d] 582 features reduced to 13 features.

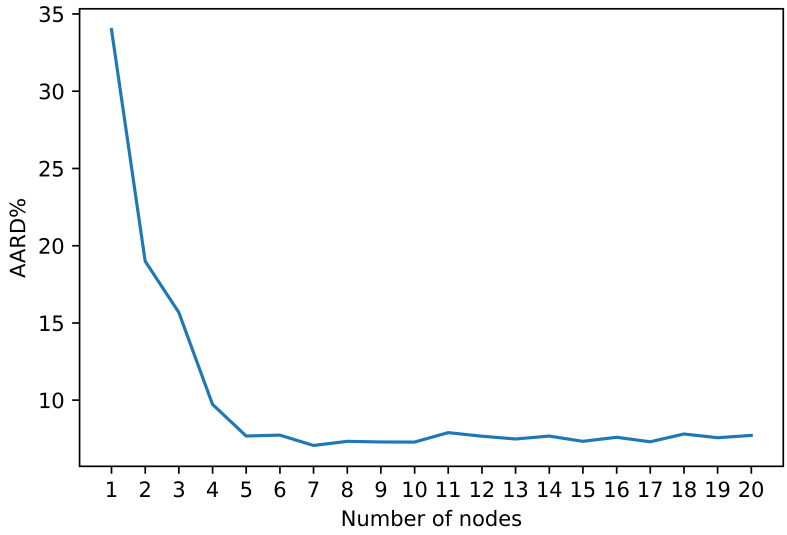


Fig. 4.8. AARDs produced with the sequential adding of nodes to the hidden layer of the FFNN, indicating a minimum error at seven nodes.

Table 4.7. Hyperparameters tested using grid-search cross-validation. The hyperparameters used in the final model are indicated in bold.

Hyperparameter	Assessed parameters
Batch Size	**10** ; 20 ; 40 ; 60 ; 80 ; 100
Optimiser	SGD ; RMSProp ; **Adam** ; Adagrad
Activation Function	tanh ; Sigmoid ; **ReLU** ; Softmax

4.4 Model Comparison

It would seem logical to directly compare the AARDs produced by the models developed in this work to determine which of these models most accurately predicts C_P. Comparing the results presented in Table 4.8, it is clear that the FFNN produced an AARD considerably lower than any of the linear models; however, the procedure followed regarding the split of data into training and testing sets differed for the linear and FFNN models, and the effect of doing so needs to be considered. This circles back to an issue raised earlier—the error of a model can be highly dependent on the data it is tested on, and it is possible that the FFNN produces low AARDs as a result of the train:test split used. To allow for a fair comparison between the performance of the linear models and the FFNN, all linear

models developed (excluding GIPF-1), using the average coefficients determined, and the FFNN, using an average over 30 predictions, are applied to the external set described in Section 3.6. To contextualise the errors produced by the models in terms of previous work, the linear model presented by Kang et al.[46] was also applied to the external data set and the AARDs produced are included in Table 4.9.

The BL model uses the same correlation (but with different fitting coefficients) presented by Preiss et al.,[21] differing only by the inclusion of a linear temperature-dependent term. The model of Preiss et al. resulted in errors of $12.38\,\%$ and $13.79\,\%$ for the prediction of C_P for [N_{4444}][doc] at $298\,\mathrm{K}$ and $323\,\mathrm{K}$, respectively; whereas the BL model produces an AARD of $4.52{\pm}2.05\,\%$ across the two temperatures. The Preiss et al. model was also applied to the [$C_nC_1$im][sac] ILs by the authors of this work and an error of approximately $12\,\%$ resulted,[156] and when the BL model was applied to the same ILs here, the AARD produced was $7.79 \pm 2.30\,\%$. The BL model therefore produces significantly more accurate results than the Preiss et al. model, indicating the effect of the training procedure used. The improved prediction could also be a result of the calculation method used to determine V_m.

The SEP model showed an improved performance compared to the BL model during model development, and an even more notable improvement results when applied to the external set. However, the results of this model are far less consistent when applied to the [sac]$^-$ series of ILs, indicating a drawback of this model in its ability to predict the effects of changing the alkyl chain length of a species.

The GIPF-2 model produced a significantly lower error than the other models, which again demonstrates the power of interacting (product) features, producing far more consistent errors than the previous models.

A clear issue with the QSPR model is that Mordred was unable to calculate the VR1_A and LogEE_A features for the [doc]$^-$ anion, and therefore cannot predict a C_P value for ILs containing this ion. The inability to calculate these features for every species indicates

Table 4.9. AARD% by produced by models when applied to the external data set. AARDs given as the average over each IL and the set overall.[a]

IL	BL	SEP	GIPF-2	QSPR	FFNN	Kang
$[C_4apy][NTf_2]$	2.35 ± 1.00	3.97 ± 1.10	1.04 ± 0.82	2.68 ± 1.66	2.25 ± 1.41	31.72 ± 1.56
$[N_{4444}][doc]$	4.52 ± 2.05	4.27 ± 2.04	7.20 ± 1.45	NA	3.47 ± 1.92	54.12 ± 4.22
$[C_8C_1pip][NTf_2]$	3.65 ± 2.11	5.14 ± 3.81	2.50 ± 1.66	23.39 ± 4.96	2.90 ± 1.77	26.07 ± 7.41
$C_8iquin][SCN]$	7.21 ± 0.84	3.85 ± 0.97	1.54 ± 0.82	33.53 ± 1.33	1.28 ± 0.69	86.18 ± 1.84
$[C_4C_1im][sac]$	7.92 ± 0.15	7.36 ± 0.19	2.37 ± 0.18	16.56 ± 1.03	5.17 ± 0.17	20.87 ± 1.16
$[C_6C_1im][sac]$	9.09 ± 0.07	9.03 ± 0.10	4.35 ± 0.91	5.17 ± 0.38	4.09 ± 0.32	30.96 ± 1.39
$[C_8C_1im][sac]$	9.07 ± 0.04	10.46 ± 0.08	3.90 ± 1.27	14.65 ± 0.15	0.92 ± 0.55	20.14 ± 1.11
$[C_{10}C_1im][sac]$	9.87 ± 0.29	4.33 ± 0.36	4.17 ± 0.25	24.48 ± 0.45	7.40 ± 0.17	157.61 ± 4.10
Overall Error	7.60 ± 2.70	6.87 ± 2.70	3.15 ± 1.53	14.39 ± 8.97	3.84 ± 2.37	53.36 ± 51.92

[a] Numbers are reported with standard deviations shown.

that such a model is not generally applicable, and the AARD produced for the rest of the external set ($14.39 \pm 8.97\,\%$) is notably larger than the AARD produced during model development ($7.36 \pm 1.77\,\%$), further showing that this model is not as robust as the other models developed.

The AARDs produced by the FFNN are within the range of the errors presented in the model development. However, during training the AARD of the FFNN was significantly lower than that of the GIPF-2 model, whereas when applied to the external set, these two models give a similar error. The comparable errors indicate that although simpler, the GIPF-2 model can provide as accurate a prediction as the FFNN.

One of the most notable results is the poor performance of the Kang model. The AARD reported by the authors was $2.51\,\%$; however, when applied to the external set, the AARD is $52.38\,\%$, indicating the sensitivity of the Kang model to the data it is trained with and tested against. All other models produced errors within reasonable agreement of the AARDs reported during model development, providing further evidence of the robustness of the training approaches used in this work.

Chapter 5

Conclusion

The project aimed to use machine-learning methods to develop generally applicable models for the prediction of ionic liquid (IL) isobaric heat capacities, C_P. A database of 2463 temperature-dependent C_P values, spanning 208 ILs, was compiled from ILThermo, for which the component ion structures were generated. Molecular volumes, V_m, features based on the electrostatic potential (ESP) and other molecular descriptors were calculated for each of the ion structures, and used to build linear and nonlinear predictive models.

Development of the baseline (BL) model investigated three methods for the determination of V_m, and it was found that the empirically determined atom and bond contribution (ABC) volumes provide the strongest correlation with C_P and were therefore used throughout the work. Temperature dependence was also explored during development of the BL model, in which it was shown that using a linear function is sufficient in describing the data set as a whole. The final BL model produced an average absolute relative deviation (AARD) of $7.28 \pm 2.08\,\%$. The inclusion of ESP-based features lowers the overall AARD of the BL model, and the inclusion of product features of V_m, temperature, and general interaction properties function (GIPF) features resulted in the most accurate linear model (the GIPF-2 model), likely due to the ability of interacting features to broadly account for cation-anion interactions, producing an AARD of $6.35 \pm 1.94\,\%$. All ESP-based

models make use of V_{m}, which makes the largest contribution to the predicted value in both the BL and SEP models and is present in the majority of the features in the GIPF-2 model, providing further evidence of the importance of volume in the description of C_P.

The FFNN was developed using the GIPF features, temperature and V_{m}, producing the lowest AARD ($2.48 \pm 3.49\,\%$) during model development, although the train and test procedure was different from that used with other models (averaging was done over the same test:train split, rather than many different splits). Due to the complex nature of a FFNN it is impossible to interpret which features make the largest contribution to the predicted value, presenting a drawback of using a FFNN for property prediction, especially where it is to be used in a rational design scheme where the inverse relationship (i.e., structure from property) is required.

All models were applied to the external data set to assess model performance on species containing "new" ions, as well as for a fair comparison between the performance of the linear models and the FFNN. The average coefficients determined during model development were used for the linear models, and the final FFNN prediction was taken as the average over 30 FFNN predictions. The linear models produced similar AARDs as seen during model development, except for the QSPR model, for which the errors were much larger. The QSPR model was unable to calculate two features for the $[\mathrm{doc}]^-$ anion, highlighting that this methodology cannot be generally applied. The GIPF-2 model and FFNN again produced the lowest errors of $3.15 \pm 1.53\,\%$ and $3.84 \pm 2.37\,\%$, respectively. The model of Kang et al.[109] was applied to the external set, producing errors notably higher than that reported by the authors. The consistency of the models in this work, compared to that of the Kang model, when applied to the external set illustrates the ability of the training procedure used here to produce robust and generally applicable models.

Overall, this work has demonstrated the predictive power of V_{m} and GIPFs, and the effect of using interacting cation-anion features for the prediction of C_P. The training procedure developed here has identified a way to develop linear regression models that

are less sensitive to the particular choice of train:test split producing errors that better represent the general prediction quality; however, this does make it difficult to individually analyse data points responsible for large errors, highlighting the necessity of an external testing set.

Building on the success of using the GIPF descriptors, future work would be to develop models incorporating these features for other properties that are key in rational design of ILs, such as viscosity, conductivity and melting point. Focus should be on improving prediction errors that are thought to result from neglecting the cation-anion interaction.

Bibliography

1. P. Walden, *Bull. Acad. Imper. Sci.*, 1914, **1800**, 405—422.

2. F. H. Hurley and T. P. Wier Jr., *J. Electrochem. Soc.*, 1951, **98**, 207—212.

3. W. T. Ford, R. J. Hauri and S. G. Smith, *J. Am. Chem. Soc.*, 1974, **96**, 4316—4318.

4. J. C. Nardi, C. L. Hussey and L. A. King, *US Pat.*, 4 122 245, 1978.

5. J. Robinson and R. A. Osteryoung, *J. Am. Chem. Soc.*, 1979, **101**, 323–327.

6. J. S. Wilkes, J. A. Levisky, R. A. Wilson and C. L. Hussey, *Inorg. Chem.*, 1982, **21**, 1263—1264.

7. E. I. Cooper and C. A. Angell, *Solid State Ion.*, 1983, **9**, 617–622.

8. E. I. Cooper and C. A. Angell, *Solid State Ion.*, 1986, **18**, 570–576.

9. J. S. Wilkes and M. J. Zaworotko, *J. Chem. Soc. Chem. Commun.*, 1982, **13**, 965—-966.

10. M. G. Freire, C. M. S. S. Neves, I. M. Marrucho, J. A. P. Coutinho and A. M. Fernandes, *J. Phys. Chem. A*, 2010, **114**, 3744—-3749.

11. T. Welton, *Chem. Rev.*, 1999, **99**, 2071–2083.

12. J. P. Hallet and T. Welton, *Chem. Rev.*, 2011, **111**, 3508–3576.

13. A. George, A. Brandt, K. Tran, S. M. S. N. S. Zahari, D. Klein-Marcuschamer, N. Sun, N. Sathitsuksanoh, J. Shi, V. Stavila, R. Parthasarathi, S. Singh, B. M. Holmes, T. Welton, B. A. Simmons and J. P. Hallett, *Green Chem.*, 2015, **17**, 1728–1734.

14. M. Watanabe, M. L. Thomas, S. Zhang, K. Ueno, T. Yasuda and K. Dokko, *Chem. Rev.*, 2017, **117**, 7190–7239.

15. K. Wang, H. Adidharma, M. Radosz, P. Wan, X. Xu, C. K. Russell, H. Tian, M. Fan and J. Yu, *Green Chem.*, 2017, **19**, 4469–4493.

16. P. Wasserscheid and W. Keim, *Angew. Chem., Int. Ed.*, 2000, **39**, 3773–3789.

17. S. Chowdhury, R. S. Mohan and J. L. Scott, *Tetrahedron*, 2007, **63**, 2363–2389.

18. T. Welton, *Coord. Chem. Rev.*, 2004, **248**, 2459–2477.

19. D. R. MacFarlane, M. Kar and J. M. Pringle, *Fundamentals of Ionic Liquids*, Wiley-VCH, Weinheim, 2017.

20. K. N. Marsh, J. A. Boxall and R. Lichtenthaler, *Fluid Ph. Equilibria*, 2004, **219**, 93–98.

21. U. P. R. M. Preiss, J. M. Slattery and I. Krossing, *Ind. Eng. Chem. Res.*, 2009, **48**, 2290—-2296.

22. J. O. Valderrama, A. Toro and R. E. Rojas, *J. Chem. Thermodyn.*, 2011, **43**, 1068–1073.

23. J. O. Valderrama, G. Martinez and C. A. Faúndez, *Int. J. Thermophys.*, 2011, **32**, 942–956.

24. M. Kar, N. V. Plechkova, K. R. Seddon, J. M. Pringle and D. R. MacFarlane, *Aust. J. Chem.*, 2019, **72**, 3–10.

25. A. S. Amarasekara, *Chem. Rev.*, 2016, **116**, 6133–6183.

26. M. J. Earle and K. R. Seddon, *Pure Appl. Chem.*, 2000, **72**, 1391–1398.

27. E. Paulechka, T. Liavitskaya and A. V. Blokhin, *J. Chem. Thermodyn.*, 2016, **102**, 211–218.

28. D. Rooney, J. Jacquemin and R. Gardas, *Top. Curr. Chem.*, 2009, **290**, 185–212.

29. I. López-Martin, E. Burello, P. N. Davey, K. R. Seddon and G. Rothenberg, *ChemPhysChem*, 2007, **8**, 690–695.

30. A. Triolo, O. Russina, H.-J. Bleif and E. Di Cola, *J. Phys. Chem. B*, 2007, **111**, 4641–4644.

31. J. N. A. Canongia Lopes and A. A. H. Pádua, *J. Phys. Chem. B*, 2006, **110**, 3330–3335.

32. R. Hayes, G. G. Warr, and R. Atkin, *Chem. Rev.*, 2015, **115**, 6357–6426.

33. P. Wasserscheid and T. Welton, *Ionic Liquids in Synthesis*, Wiley-VCH, Weinheim, 2008.

34. X. Wang, Y. Chi and T. Mu, *J. Mol. Liq.*, 2014, **193**, 262–266.

35. L. M. Diamante and T. Lan, *J. Food Process. Preserv.*, 2014, **2014**, 1–6.

36. M. Ngadi and L. Yu, *Can. Biosyst. Eng.*, 2004, **46**, 315–318.

37. O. Ivanciuc, T. Ivanciuc, P. A. Filip and D. Cabrol-Bass, *J. Chem. Inf. Comput. Sci.*, 1999, **39**, 515–524.

38. R. N. Das and K. Roy, *Mol. Divers.*, 2013, **17**, 151–196.

39. M. J. Earle, J. M. S. S. Esperanca, M. A. Gilea, J. N. C. Lopes, L. P. Rebelo, J. W. Magee, K. R. Seddon and J. A. Widegren, *Nature*, 2006, **439**, 831–834.

40. M. Armand, F. Endres, D. R. MacFarlane, H. Ohno and B. Scrosati, *Nat. Mater.*, 2009, **8**, 621–629.

41. S. P. M. Ventura, F. A. e Silva, M. V. Quental, D. Mondal, M. G. Freire and J. A. P. Coutinho, *Chem. Rev.*, 2017, **117**, 6984–7052.

42. V. I. Parîvulescu and C. Hardacre, *Chem. Rev.*, 2007, **107**, 2615–2665.

43. J. E. Bara, T. K. Carlisle, C. J. Gabriel, D. Camper, A. Finotello, D. L. Gin and R. D. Noble, *Ind. Eng. Chem. Res.*, 2009, **48**, 2739—2751.

44. G. A. Giffin, *J. Mater. Chem. A*, 2016, **4**, 13378–13389.

45. D. R. MacFarlane, N. Tachikawa, M. Forsyth, J. M. Pringle, P. C. Howlett, G. D. Elliott, J. H. Davis, M. Watanabe, P. Simon and C. A. Angell, *Energy Environ. Sci.*, 2014, **7**, 232–250.

46. X. Kang, X. Sun and B. Han, *Adv. Mater.*, 2016, **28**, 1011–1030.

47. M. Zábranský, V. Růžička Jr. and E. S. Domalski, *J. Phys. Chem. Ref. Data*, 2001, **30**, 1199–1689.

48. J. S. Rowlinson and F. L. Swinton, in *Liquids and Liquid Mixtures*, ed. J. S. Rowlinson and F. L. Swinton, Butterworth Scientific, London, 3rd edn, 1982, ch. 2, pp. 11–58.

49. M. E. Van Valkenburg, R. L. Vaughn, M. Williams and J. S. Wilkes, *Thermochim. Acta*, 2005, **425**, 181–188.

50. E. Wilhelm and T. M. Letcher, *Heat Capacities Liquids, Solutions and Vapours*, The Royal Society of Chemistry, Cambridge, 2010.

51. B. E. Poling, J. M. Prausnitz and J. P. O'Connell, *The Properties of Gases and Liquids*, McGraw-Hill, New York, 5th edn, 2001.

52. K. G. Joback and R. C. Reid, *Chem. Eng. Commun.*, 1987, **57**, 233–243.

53. V. Růžička Jr. and E. S. Domalski, *J. Phys. Chem. Ref. Data*, 1993, **22**, 597–618.

54. V. Růžička Jr. and E. S. Domalski, *J. Phys. Chem. Ref. Data*, 1993, **22**, 619–657.

55. S. W. Benson and J. H. Buss, *J. Chem. Phys.*, 1958, **29**, 546–572.

56. S. W. Benson, F. R. Cruickshank, D. M. Golden, G. R. Haugen, H. E. O'Neal, A. S. Rodgers, R. Shaw and R. Walsh, *Chem. Rev.*, 1969, **69**, 279–324.

57. M. Luria and S. W. Benson, *J. Chem. Eng. Data*, 1977, **22**, 90–100.

58. A. C. Mater and M. L. Coote, *J. Chem. Inf. Model.*, 2019, **59**, 2545–2559.

59. J. A. P. Coutinho, P. J. Carvalho and N. M. C. Oliveira, *RSC Adv.*, 2012, **2**, 7322–7346.

60. Z. Wu, B. Ramsundar, E. N. Feinberg, J. Gomes, C. Geniesse, A. S. Pappu, K. Leswing and V. Pande, *Chem. Sci.*, 2018, **9**, 513–530.

61. A. Lydersen, *Estimation of Critical Properties of Organic Compounds by the Method of Group Contributions*, Engineering Experiment Station Report 3, University of Wisconsin, Madison, 1955.

62. K. G. Joback, *MSc thesis*, Massachusetts Institute of Technology, 1984.

63. K. M. Kleincewicz and R. C. Reid, *AlChE J.*, 1984, **30**, 137–142.

64. L. Constantinou and R. Gani, *AlChE J.*, 1994, **40**, 1697–1710.

65. V. Růžička Jr., M. Zábranský, A. Malijevský and E. S. Domalski, *Fluid Phase Equilib.*, 1992, **75**, 137–148.

66. D. Waliszewski, I. Stępniak, H. Piekarski and A. Lewandowski, *Thermochim. Acta*, 2005, **433**, 149–152.

67. C. F. Chueh and A. C. Swanson, *Can. J. Chem. Eng.*, 1973, **51**, 596–600.

68. R. Ge, C. Hardacre, J. Jacquemin, P. Nancarrow and D. W. Rooney, *J. Chem. Eng. Data*, 2008, **53**, 2148–2153.

69. J. O. Valderrama and P. A. Robles, *Ind. Eng. Chem. Res.*, 2007, **46**, 1338–1344.

70. R. L. Gardas, R. Ge, P. Goodrich, C. Hardacre, A. Hussain and D. W. Rooney, *J. Chem. Eng. Data*, 2009, **55**, 1505–1515.

71. J. Albert and K. Müller, *Ind. Eng. Chem. Res.*, 2014, **53**, 17522–17526.

72. R. L. Gardas and J. A. P. Coutinho, *Ind. Eng. Chem. Res.*, 2008, **47**, 5751–5757.

73. M. Zábranský and V. R. Jr., *J. Phys. Chem. Ref. Data*, 2004, **33**, 1071–1081.

74. A. N. Soriano, A. M. Agapito, L. J. L. I. Lagumbay, A. R. Caparanga and M. Li, *J. Taiwan Inst. Chem. Eng.*, 2010, **41**, 307–314.

75. C. J. Cramer, *Essentials of Computational Chemistry: Theories and Models*, Wiley, Chichester, 2nd edn, 2004.

76. K. Müller and J. Albert, *Ind. Eng. Chem. Res.*, 2014, **53**, 10343–10346.

77. P. Nancarrow, M. Lewis and L. AbouChacra, *Chem. Eng. Technol.*, 2015, **38**, 632–644.

78. Y. Chen, G. M. Kontogeorgis and J. M. Woodley, *Ind. Eng. Chem. Res.*, 2019, **58**, 4277–4292.

79. J. O. Valderrama and R. E. Rojas, *Fluid Ph. Equilibria*, 2010, **297**, 107–112.

80. M. Randić, *J. Am. Chem. Soc.*, 1975, **97**, 6609–6615.

81. R. Todeschini and V. Consonni, *Handbook of Molecular Descriptors*, Wiley-VCH, Weinheim, 2000.

82. N. Farahani, F. Gharagheizi, S. A. Mirkhani and K. Tumba, *Fluid Ph. Equilibria*, 2013, **337**, 73–82.

83. M. Sattari, F. Gharagheizi, P. Ilani-Kashkouli, A. H. Mohammadi and D. Ramjugernath, *Ind. Eng. Chem. Res.*, 2013, **52**, 13217–13221.

84. M. Sattari, F. Gharagheizi, P. Ilani-Kashkouli, A. H. Mohammadi and D. Ramjugernath, *J. Therm. Anal. Calorim.*, 2014, **115**, 1863–1882.

85. A. Ahmadi, R. Haghbakhsh, S. Raeissi and V. Hemmati, *Fluid Ph. Equilibria*, 2015, **403**, 95–103.

86. A. Paternò, R. Fiorenza, S. Marullo, G. Musumarra and S. Scirè, *RSC Adv.*, 2016, **6**, 36085—-36089.

87. G. Cruciania, P. Crivorib, P. Carrupt and B. Testa, *J. Mol. Struct. (THEOCHEM)*, 2000, **503**, 17–30.

88. W. He, F. Yan, Q. Jia, S. Xia and Q. Wang, *Fluid Ph. Equilibria*, 2019, **500**, 112260.

89. L. Glasser and H. D. Jenkins, *Chem. Soc. Rev.*, 2005, **34**, 866–874.

90. L. Glasser and H. D. B. Jenkins, *Thermochim. Acta*, 2004, **414**, 125–130.

91. H. D. B. Jenkins and L. Glasser, *Inorg. Chem.*, 2003, **42**, 8702–8708.

92. H. D. B. Jenkins, D. Tudela and L. Glasser, *Inorg. Chem.*, 2002, **41**, 2364–2367.

93. L. Glasser and H. D. B. Jenkins, *J. Am. Chem. Soc.*, 2000, **122**, 632–638.

94. H. D. B. Jenkins, H. K. Roobottom, J. Passmore and L. Glasser, *Inorg. Chem.*, 1999, **38**, 3609–3620.

95. T. E. Mallouk, G. L. Rosenthal, G. Mueller, R. Brusasco and N. Bartlett, *Inorg. Chem.*, 1984, **23**, 3167–3173.

96. L. Glasser and H. D. B. Jenkins, *Inorg. Chem.*, 2008, **47**, 6195–6202.

97. H. D. B. Jenkins and J. F. Liebman, *Inorg. Chem.*, 2005, **44**, 6359–6372.

98. L. Glasser and H. D. B. Jenkins, *Inorg. Chem.*, 2011, **50**, 8565–8569.

99. L. Glasser and H. D. B. Jenkins, *Phys. Chem. Chem. Phys.*, 2016, **18**, 21226–21240.

100. A. A. Strechan, A. G. Kabo, Y. U. Paulechka, A. V. Blokhin, G. J. Kabo, A. S. Shaplov and E. I. Lozinskaya, *Thermochim. Acta*, 2008, **474**, 25–31.

101. Y. U. Paulechka, A. G. Kabo, A. V. Blokhin, G. J. Kabo and M. P. Shevelyova, *J. Chem. Eng. Data*, 2010, **55**, 2719–2724.

102. A. Klamt, *WIREs Comput. Mol. Sci.*, 2011, **1**, 699–709.

103. I. Krossing, J. M. Slattery, C. Daguenet, P. J. Dyson, A. Oleinikova and H. Weingärtner, *J. Am. Chem. Soc.*, 2006, **128**, 13427–13434.

104. J. M. Slattery, C. Daguenet, P. J. Dyson, T. J. S. Schubert and I. Krossing, *Angew. Chem. Int. Ed.*, 2007, **46**, 5384–5388.

105. U. P. R. M. Preiss, S. P. Verevkin, T. Koslowski and I. Krossing, *Chem. Eur. J.*, 2011, **17**, 6508–6517.

106. D. H. Zaitsau, G. J. Kabo, A. A. Strechan, Y. U. Paulechka, A. Tschersich, S. P. Verevkin and A. Heintz, *J. Phys. Chem. A*, 2006, **110**, 7303–7306.

107. W. Beichel, U. P. R. M. Preiss, S. P. Verevkin, T. Koslowski and I. Krossing, *J. Mol. Liq.*, 2014, **192**, 3–8.

108. Y. Zhao, S. Zeng, Y. Huang, R. M. Afzal and X. Zhang, *Ind. Eng. Chem. Res.*, 2015, **54**, 12987–12992.

109. X. Kang, X. Liu, J. Li, Y. Zhao and H. Zhang, *Ind. Eng. Chem. Res.*, 2018, **57**, 16989–16994.

110. A. Barati-Harooni, A. Najafi-Marghmaleki and A. H. Mohammadi, *J. Mol. Liq.*, 2017, **227**, 324–332.

111. J. R. Elliott and C. T. Lira, *Introductory Chemical Engineering Thermodynamics*, Pearson Education, Inc, New Jersey, 2nd edn, 2012.

112. R. S. Benson, *Advanced Engineering Thermodynamics*, Pergamon Press, 2nd edn, 1977.

113. L. P. Rebelo, V. Najdanovic-Visak, R. Gomes de Azevedo, J. M. S. S. Esperança, M. Nunes da Ponte, H. J. R. Guedes, Z. P. Visak, H. C. de Sousa, J. Szydlowski, J. N. Canongia Lopes and T. C. Cordeiro, in *Phase behavior and thermodynamic properties of ionic liquids, ionic liquid mixtures, and ionic liquid solutions*, ed. R. D. Rogers and K. R. Seddon, ACS Publishing, Washington, 2005, vol. 901, book section 21, pp. 270–291.

114. A. Géron, *Hands-on Machine Learning with Scikit-Learn, Keras & TensorFlow*, O'Reilly Media, Inc., Sebastopol, 2nd edn, 2019.

115. C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, New York, 2006.

116. R. Tibshirani, *J. R. Stat. Soc . Series B Stat. Methodol.*, 1996, **58**, 267–288.

117. A. E. Hoerl and R. W. Kennard, *Technometrics*, 1970, **12**, 55–67.

118. F. Jensen, *Introduction to Computational Chemistry*, John Wiley and Sons Ltd, Sussex, 2006.

119. R. J. Bartlett and M. Musiał, *Rev. Mod. Phys.*, 2007, **79**, 291–352.

120. P. Hohenberg and W. Kohn, *Phys. Rev.*, 1964, **136**, B864–B871.

121. N. Mardirossian and M. Head-Gordon, *Mol. Phys.*, 2017, **115**, 2315–2372.

122. A. Kazakov, J. W. Magee, R. D. Chirico, E. Paulechka, V. Diky, C. D. Muzny, K. Kroenlein and M. Frenkel, *"NIST Standard Reference Database 147: NIST Ionic Liquids Database - (ILThermo)", Version 2.0, National Institute of Standards and Technology, Gaithersburg MD, 20899*, `https://ilthermo.boulder.nist.gov`, (accessed June 2019).

123. Q. Dong, C. D. Muzny, A. Kazakov, V. Diky, J. W. Magee, J. A. Widegren, R. D. Chirico, K. N. Marsh and M. Frenkel, *J. Chem. Eng. Data*, 2007, **52**, 1151–1159.

124. K. Seddon, A. Stark and M. J. Torres, *Pure Appl. Chem.*, 2000, **72**, 2275–2287.

125. S. Grimme, C. Bannwarth and P. Shushkov, *J. Chem. Theory Comput.*, 2019, **15**, 1652–1671.

126. J. P. Perdew, K. Burke and M. Ernzerhof, *Phys. Rev. Lett.*, 1996, **77**, 3865–3868.

127. F. Weigend and R. Ahlrichs, *Phys. Chem. Chem. Phys.*, 2005, **7**, 3297–3305.

128. J. P. Perdew, A. Ruzsinszky, V. N. Tao, J.; Staroverov, G. E. Scuseria and G. b. I. Csonka, *J. Chem. Phys.*, 2005, **123**, 062201.

129. S. Grimme, M. Steinmetz and M. Korth, *J. Org. Chem.*, 2007, **72**, 2118–2126.

130. F. Weigend, M. Häser, H. Patzelt and R. Ahlrichs, *Chem. Phys. Lett.*, 1998, **294**, 143–152.

131. F. Weigend, *Phys. Chem. Chem. Phys.*, 2006, **8**, 1057–1065.

132. F. Neese, *WIREs Comput. Mol. Sci.*, 2018, **8**, 1–6.

133. L. Glasser and H. D. B. Jenkins, *Inorg. Chem.*, 2003, **42**, 8702–8708.

134. J. L. Pascual-Ahuir, E. Silla and I. Tunon, *J. Comput. Chem.*, 1994, **15**, 1127–1138.

135. Y. Takano and K. N. Houk, *J. Chem. Theory Comput.*, 2005, **1**, 70–77.

136. R. F. W. Bader, M. T. Carroll, J. R. Cheeseman and C. Chang, *J. Am. Chem. Soc.*, 1987, **109**, 7968–7979.

137. T. Lu and F. Chen, *J. Comput. Chem.*, 2012, **33**, 580–592.

138. T. Lu and F. Chen, *J. Mol. Graph. Model.*, 2012, **38**, 314–323.

139. Y. H. Zhao, M. H. Abraham and A. M. Zissimos, *J. Org. Chem.*, 2003, **68**, 7368–7373.

140. H. Moriwaki, Y. Tian, N. Kawashita and T. Takagi, *J. Cheminform.*, 2018, **10**, 1758–2946.

141. J. S. Murray and P. Politzer, *WIREs Comput. Mol. Sci.*, 2011, **1**, 153–163.

142. J. S. Murray and P. Politzer, *Theor. Chem. Acc.*, 2002, **108**, 134–142.

143. J. S. Murray, T. Brinck, P. Lane, K. Paulsen and P. Politzer, *J. Mol. Struct. (THEOCHEM)*, 1994, **307**, 55–64.

144. J. S. Murray and P. Politzer, *J. Mol. Struct. (THEOCHEM)*, 1998, **425**, 107–114.

145. J. S. Murray, T. Brinck and P. Politzer, *Chem. Phys.*, 1996, **204**, 289–299.

146. *RDKit: Open-source cheminformatics*, `http://www.rdkit.org`, (accessed May 2020).

147. R. M. Jarvis, D. Broadhurst, H. Johnson, N. M. O'Boyle and R. Goodacre, *Bioinformatics*, 2006, **22**, 2565—-2566.

148. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay, *J. Mach. Learn. Res.*, 2011, **12**, 2825–2830.

149. W. Beckner, C. M. Mao and J. Pfaendtner, *Mol. Syst. Des. Eng.*, 2018, **3**, 253–263.

150. F. Chollet, *Keras*, `https://keras.io`, 2015, (accessed November 2019).

151. K. Hornik, *Neural Netw.*, 1991, **4**, 251–275.

152. A. Diedrichs and J. Gmehling, *Fluid Ph. Equilibria*, 2006, **244**, 68–77.

153. J. M. Crosthwaite, M. J. Muldoon, J. K. Dixon, J. L. Anderson and J. F. Brennecke, *J. Chem. Thermodyn.*, 2005, **37**, 559–568.

154. G. Chatel, L. Leclerc, E. Naffrechoux, C. Bas, N. Kardos, C. Goux-Henry, B. Andrioletti and M. Draye, *J. Chem. Eng. Data*, 2012, **57**, 3385–3390.

155. M. Królikowska, K. Paduszyński and M. Zawadzki, *J. Chem. Eng. Data*, 2013, **58**, 285–293.

156. M. Bendová, M. Čanji, Z. Wagner and M. G. Bogdanov, *J. Solution Chem.*, 2018, **48**, 949–961.

157. M. Zábranský, Z. Kolská, V. Růžička and E. S. Domalski, *J. Phys. Chem. Ref. Data*, 2010, **39**, 013103.

158. K. Das, I. Gutman and B. Furtula, *Filomat*, 2012, **26**, 733–738.

159. *Mordred 1.2.1a1 Documentation*, `https://mordred-descriptor.github.io/documentation/master/descriptors.html`, (accessed March 2020).

160. J. Gasteiger and M. Marsili, *Tetrahedron Lett.*, 1978, **19**, 3181–3184.

161. L. H. Hall and L. B. Kier, in *The Molecular Connectivity Chi Indexes and Kappa Shape Indexes in Structure-Property Modeling*, ed. K. B. Lipkowitz and D. B. Boyd, Wiley-VCH, Weinheim, 1991, vol. 2, book section 9, pp. 367–422.

Appendix A

Linear Regression

```
def reg_model(X,y,iterations=1000, names=None,scale=True):
    import pandas as pd
    import numpy as np
    from sklearn.linear_model import LinearRegression
    from sklearn.model_selection import GroupShuffleSplit
    from sklearn import preprocessing

    # Initialise series
    reg_model.n_features = pd.Series([])
    reg_model.r2 = pd.Series([])
    reg_model.adj_r2 = pd.Series([])
    reg_model.aard = pd.Series([])
    reg_model.coefficients = pd.Series([])
    reg_model.intercept = pd.Series([])

    # Create regressor
    reg = LinearRegression()

    # Create scaler
    scaler = preprocessing.StandardScaler()

```

```
    # Perform test/train split specified amount of times
    for i in range(iterations):
        train_inds, test_inds = next(GroupShuffleSplit(test_size=.10,
n_splits=2, random_state = i).split(X,y, groups=names))
        try:
            x_train = X.iloc[train_inds]
            x_test = X.iloc[test_inds]
        except:
            x_train = X[train_inds]
            x_test = X[test_inds]

        y_train = y.iloc[train_inds].values
        y_test = y.iloc[test_inds].values

        # Scale features
        if scale == True:
            X_train = scaler.fit_transform(x_train)
            X_test = scaler.transform(x_test)
        else:
            X_train = x_train
            X_test = x_test

        # Fit model
        model = reg.fit(X_train,y_train)
        y_pred = reg.predict(X_test)
        resid = y_test - y_pred
        ard = (resid/y_test)*100
        mod_ard = abs(ard)
        reg_model.aard[i] = mod_ard.mean()
        reg_model.r2[i] = r2_score(y_test,y_pred)
        reg_model.n_features[i] = np.sum(model.coef_!=0)
        reg_model.coefficients[i] = np.array(model.coef_)
        reg_model.intercept[i] = model.intercept_
```

```
54
55          # Calculate adjusted R-squared
56          try:
57              reg_model.adj_r2[i] = 1- (((1-reg_model.r2[i])*(len(X_test)-1)
    )/(len(X_test)-len(X.columns) - 1))
58          except:
59              reg_model.adj_r2[i] = 1- (((1-reg_model.r2[i])*(len(X_test)-1)
    )/(len(X_test)-len(X[0]) - 1))
60
61      # Coefficients from each fitting iteration
62      reg_model.coefficients = reg_model.coefficients.transpose()
63
64      # General model output
65      print("Intercept = ",reg_model.intercept.mean(),"+-", stdev(reg_model.
    intercept))
66      print("AARD% = ",reg_model.aard.mean(), "+-", stdev(reg_model.aard))
67      print("R-squared = ",reg_model.r2.mean(),"+-",stdev(reg_model.r2))
68      print("R-squared (adj) = ",reg_model.adj_r2.mean(),"+-",stdev(
    reg_model.adj_r2))
```

Appendix B

Lasso Regularisation

```
import numpy as np
import pandas as pd
import datetime
from sklearn import preprocessing
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import r2_score
from sklearn.model_selection import GroupShuffleSplit
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LassoCV
import multiprocessing
import concurrent.futures
import os

def reg_model_2(i):
    # Time function call
    start = datetime.datetime.now()

    # Import data
    dataset = pd.read_csv("Areas_updated3.csv")
    names = dataset["Unnamed: 0"]

```

```
    # Test/train split
    print("Train/Test split ", i + 1)
    train_inds, test_inds = next(GroupShuffleSplit(test_size=.10, n_splits
=2, random_state = i).split(dataset, groups=names))
    try:
        train = dataset.iloc[train_inds]
        test = dataset.iloc[test_inds]
    except:
        train = dataset[train_inds]
        test = dataset[test_inds]

    try:
        names_in_training_set = names.iloc[train_inds]
    except:
        names_in_training_set = names[train_inds]

    x_train = train.iloc[:,3:]
    y_train = train["Heat capacity"]
    x_test = test.iloc[:,3:]
    y_test = test["Heat capacity"]

    # Create scaler
    scaler = preprocessing.StandardScaler()

    # Transform the features
    X_train = scaler.fit_transform(x_train)
    X_test = scaler.transform(x_test)

    # Fit LASSO
    alphas_list = np.logspace(-2, 2, 300)
    splitter = GroupShuffleSplit(test_size=.20, n_splits=5, random_state =
    i).split(X_train, groups=names_in_training_set)
    reg = LassoCV(cv=iter(splitter), alphas=alphas_list, max_iter
```

```
    =100000000, n_jobs=None)
    reg.fit(X_train, y_train)

    y_pred = reg.predict(X_test)
    aard = abs(((y_test - y_pred)/y_test)*100).mean()
    r2 = r2_score(y_test, y_pred)
    n_feats = np.sum(reg.coef_!=0)
    coeffs = np.array(reg.coef_)
    ints = reg.intercept_
    alpha = reg.alpha_

    print(f'AARD: {aard}')
    print(f'R^2:  {r2}')
    print(f'Intercept: {ints}')
    print(f'# features: {n_feats}')
    print(f'alpha: {alpha}')

    # Time function call
    print(f'Run time: {datetime.datetime.now() - start}')

    return (aard, r2, coeffs, ints, n_feats, alpha)


if __name__ == '__main__':
    start = datetime.datetime.now()
    df = pd.DataFrame()
    aards = pd.Series([])
    r_squared = pd.Series([])
    coefficients = pd.Series([])
    intercepts = pd.Series([])
    n_features = pd.Series([])
    alphas = pd.Series([])

```

```
    # Set number of CPUs
    cpus = multiprocessing.cpu_count()
    # Set threads to 1 in MKL calls
    os.environ["OMP_NUM_THREADS"] = "1"
    os.environ["MKL_NUM_THREADS"] = "1"
    n = 0
    with concurrent.futures.ProcessPoolExecutor(max_workers=cpus) as
executor:
        future_split = {executor.submit(reg_model_2, i): i for i in range
(1000)}
        for future in concurrent.futures.as_completed(future_split):
            test = future_split[future]
            aard, r2, coeffs, ints, n_feats, alpha = future.result()
            n += 1
            aards[n] = aard
            r_squared[n] = r2
            coefficients[n] = coeffs
            intercepts[n] = ints
            n_features[n] = n_feats
            alphas[n] = alpha

    df["AARD"] = aards
    df["R-squared"] = r_squared
    df["Coefficients"] = coefficients
    df["Intercept"] = intercepts
    df["Average number of features"] = n_features
    df["Alpha"] = alphas
    df.to_csv("kang3.csv")
    print(f'AARD%: {aards.mean()}')
    print(f'Total run time: {datetime.datetime.now() - start}')
```

Appendix C

Feed Forward Neural Network

```
import numpy as np
np.random.seed(19950612)
import pandas as pd
from sklearn import preprocessing
from sklearn.model_selection import GroupShuffleSplit
from tensorflow.python.keras import models
from tensorflow.python.keras import layers
from tensorflow.python.keras.callbacks import EarlyStopping,
    ModelCheckpoint
from sklearn.metrics import mean_absolute_error
from statistics import stdev

mae = pd.Series([])
aard = pd.Series([])
bias0 = pd.Series([])
bias1 = pd.Series([])

# Import stratified dataset
data = pd.read_csv("stratified_esp3.csv")

# Split into test and train set
```

```
train = data[data["Split"] == "train"]
test = data[data["Split"] == "test"]
val = data[data["Split"]=="val"]
x_train = train.iloc[:,4:]
x_test = test.iloc[:,4:]
x_val = val.iloc[:,4:]
target_train = train["Heat Capacity"].values
target_test = test["Heat Capacity"].values
target_val = val["Heat Capacity"].values

# Scale data
scaler = preprocessing.StandardScaler()
features_train = scaler.fit_transform(x_train)
features_test = scaler.transform(x_test)
features_val = scaler.transform(x_val)

# Specify parameters
h_nodes = 7 # set number of hidden nodes
early_stopping = True # early stopping switched on
batch_size = 10 # set batch size
activation = "relu" # set activation function
optimizer = "rmsprop" # set optimiser

#Fit neural network 30 times
for i in range(30):

    # Start neural network
    network = models.Sequential()

    # Add first hidden layer
    network.add(layers.Dense(units=h_nodes,
                             activation=activation,
                             input_shape=(features_train.shape[1],)))
```

```
54
55      # Add output layer
56      network.add(layers.Dense(units=1))
57
58      # Compile neural network
59      network.compile(loss="mse",
60                      optimizer=optimizer,
61                      metrics=["mse"])
62
63      # Call back used for early stopping
64      if early_stopping == True:
65          callbacks = [EarlyStopping(monitor="val_loss", patience=2),
66                       ModelCheckpoint(filepath="best_model.h5",monitor="
    val_loss",save_best_only=True)]
67      else:
68          callbacks = None
69
70      # Train neural network
71      history = network.fit(features_train, # training features
72                            target_train, # target features
73                            epochs=500, # number of training epochs
74                            callbacks=callbacks,
75                            verbose=0,
76                            batch_size=batch_size,
77                            validation_data=(features_val, target_val))
78
79      # Predicted values of test set
80      predicted_target = network.predict(features_test)
81      mae[i] = mean_absolute_error(target_test,predicted_target)
82      aard[i] = (mean_absolute_error(target_test,predicted_target)/
    target_test.mean())*100
83      print("Iteration: ", i+1 ,"; Epochs ran: ",len(history.history['loss'
    ]),"; MAE: ","%.2f" % mae[i],"; AARD%: ","%.2f" % aard[i])
```

```
84
85 print("AVERAGE ERROR = ","%.2f" % mae.mean(),"+-","%.2f" % stdev(mae))
86 print("AARD% = ","%.2f" % aard.mean(),"+-","%.2f" % stdev(aard))
```