Prediction of polymer properties using machine learning with trainless neural architecture search

March 2022

Ekaterina Gracheva

Prediction of polymer properties using machine learning with trainless neural architecture search

Graduate School of Systems and Information Engineering

University of Tsukuba

March 2022

Ekaterina Gracheva

Abstract

In the present thesis we aim to accelerate the development of novel polymers. Specifically, we try to replace expensive and time-consuming experiments for three thermal polymer properties with machine learning models. For this we choose to represent molecular structure with text strings (so-called SMILES format) and to apply techniques of machine translation from the natural language processing field. For this, we use the SMILES-X molecular prediction software. SMILES-X models translate phrases written in chemical language into the property of interest.

On the way of implementation of the model, we faced the problem of hyperparameters optimisation. In the field of neural architecture search, particular difficulty represents simultaneous determination of the geometry and training hyperparameters. In this thesis, we aim to separate the two processes by developing a fully trainless geometry search algorithm. This method also allows to determine training hyperparameters with higher confidence. When applied to the SMILES-X, trainless neural architecture search brings significant improvement. The discovered neural architectures show very good and stable results.

The property we have predicted is the coefficient of linear thermal expansion, β with 106 samples. It shows R^2-score of 68 %. This performance is very good comparing to existing methods, and additional validation with experimental sample and analysis of the interpretation maps provided by SMILES-X further prove its reliability. The model for β are ready to replace experiments, and can be further improved with more data.

Acknowledgements

I would like to express my sincere appreciation to Dr. Ayako Nakata, for supervising my work with support and trust. I also thank Dr. Keitaro Sodeyama for the chance of working at the National Institute for Materials Science during my studies. I thank Sadaki Samitsu for sharing his knowledge on polymer physics with me and valuable literature recommendations.

I am grateful to all the members of me thesis committee: Professor Tetsuya Sakurai, Professor Sadaki Samitsu, Professor Jun Sakuma, Associate Professor Youhei Akimoto, for taking their time to read, understand and comment on such an interdisciplinary dissertation.

I thank my family. My dear children, Alexander and Mila, even though you cannot read it yet, I thank you for being there for me when I needed it, and I am sorry for all the hours when I was not there for you. I thank my mother, Dr. Marina Serebryakova, for her continuous encouragement and wisdom. I thank my husband, Dr. Guillaume Lambard, for his faith, for backing me all the way through and for taking care of our children when I could not do so.

I thank my friends. Nikita Khodakovsky, Ilya Komarov, Inna Zykova, Anastasia Tautenhahn, Maria Mitrofanova, Dmitry Zadorin, Evgeniya Golovanova, Irina Guseva. Regardless the time and the distance separating us, I carry your support and your love in my heart.

Contents

List of Figures

List of Tables

1. Thesis Purpose and Structure

Polymers are long molecular chains composed of short repeating units bound together. The importance of polymers in modern world is difficult to overestimate. Their properties allow to fit a vast range of applications. Plastics replace scarce natural resources making produced goods cheaper and more accessible, so the demand on novel plastics is never ceasing. However, the development of a single material from inception to industrialisation takes on average from 15 to 25 years [1]. This slow rate is mainly due to expensive and time consuming experiments, which are required to support the trial and error method widely used in the field.

On the same time, the chemical search space is extremely vast and scarce. The upper limit on the number of potential molecules is estimated to be of the order of 10^{180} (the number of atoms in the observable Universe is estimated to be somewhere between 10^{78} and 10^{82}). The largest database of molecular compounds, Chemical Abstracts Service (CAS), contains registered information of about 200 million molecules, with only about $350,000$ out of which are registered for use today [2]. It is obvious that in order to accelerate the exploration of chemical space one needs to replace experiments with faster and cheaper estimation methods.

For crystalline materials with ordered structure (*e.g.*, metals, minerals, ceramics), experiments are often substituted with theoretical simulations based on rigorous physical laws, such as first principle calculations. Amorphous polymers, on the other hand, are composed of long disorderly entangled macromolecules, showing much more complex dynamics. Large system sizes necessary for accurate property predictions make the investigation via first principles calculations extremely computationally expensive [3].

Recently, machine learning (ML) shows promising results predicting various materials properties [4]–[8]. ML methods are suitable for large amounts of data and can be used to deduce property values by generalising over available data. They are believed to accelerate the development of materials by reducing the number of physical experiments and narrowing down the search space.

However, ML also comes with some limitations. Specifically, every ML algorithm requires time-consuming neural architecture search (NAS) step. To address this issue, in the present thesis we formulate the following aims:

1. Develop a trainless NAS technique. This technique should be computationally cheap and fast, while being precise.

2. Build reliable predictive models for several polymer thermal properties. These models should be used to substitute experiments.

The thesis is structured as follows. Chapter 2 covers background on the polymer research, introduce polymer properties of interest and discuss prediction methods existing in the field. Chapter 3 provides current status on ML in materials science. Chapters 4 and 5 explain the research on the trainless (NAS) and the prediction of polymer thermal properties, respectively. The general conclusion of the PhD research is given in Chapter 6.

2. Background: Polymers

2.1 History

The concept of polymers was first theorised in 1920 by a German chemist Hermann Staudinger [9]. He suggested that natural rubber consists of long molecular chains composed of short repeating units bound together. This hypothesis was not accepted until the direct X-ray observation made by Staudinger's Austrian namesake, Herman Mark, who is now known as the father of polymer science [10]. Experimental evidence convinced scientific society and Staudinger received a Nobel Prize in Chemistry in 1953. (Curiously, earlier Herman Mark also helped Albert Einstein to verify the Compton effect, which confirmed Einstein's quantum light theory, for which Einstein was awarded a Nobel Prize in Physics in 1921).

The term "polymer" that was coined by a Swedish chemist Jöns Jacob Berzelius almost a hundred years earlier, in 1833, originates from the Greek words $\pi o \lambda \upsilon \varsigma$ and $\mu \epsilon \rho o \varsigma$ (*polus* and *meros*) and translates as "many parts" [11]. At that time, however, the term was related to the isomerism of small molecules, and had little to do with their molecular weight. Because of the inaccurate definition given by Berzelius, the term "polymer" soon becomes confused with "isomer", and finally loses its original meaning.

Amusingly, the term ”polymer” is still being confused at present, but with a different term – that of ”macromolecules”. The reason is that Hermann Staudinger, wishing to avoid further confusion with isomerism first calls long molecular chains ”macromolecules”. Nowadays, terms ”macromolecule” and ”polymer” are often used interchangeably. However, there is a distinct difference between the two: while a polymer is a large molecule consisting of repeating units, a macromolecule is any molecule with high molecular mass. Therefore, while macromolecules may or may not be polymers, all polymers are macromolecules. The Gold Book of the International Union of Pure and Applied Chemistry [12] gives the following definition of a polymer molecule as a compromise: ”A molecule of high relative molecular mass, the structure of which *essentially* comprises the multiple repetition of units derived, actually or conceptually, from molecules of low relative molecular mass.”

As it often happens in history, polymer development and production started long before the material was theoretically described. Archaeological evidences indicate that natural rubber was used by Ancient Mesoamericans to make balls used for the Mesoamerican ballgame (1600 BC) [13]. The first scientific paper describing rubber properties was published in 1755, the modern name being given fifteen years later, in 1770, when an English chemist Joseph Priestley notices that rubber is remarkably good for rubbing off pencil marks. Rubber vulcanisation, the process that causes molecules within the material to cross-link and to form a single giant molecule, has been discovered by Charles Goodyear in 1839, who accidentally dropped a mixture of rubber and sulphur on a frying pan. In 1856 an English inventor Alexander Parkes transformed cellulose into the first man-made plastic – Parkesine. This material was later modified by John Wesley Hyatt and patented under the name ”celluloid”.

The very first fully synthetic polymer, Baleskine, was invented in 1907 by Leo Baekeland in the United States. Poly(vinyl chloride), PVC, which is by now one of the most widely produced synthetic plastic polymers, was accidentally discovered twice, by two independent researchers, Henri Victor Regnault and Eugen Baumann, in 1838 and 1872, respectively, but none of them undertook its commercialisation. It was not before 1913 that an efficient PVC production scheme was developed and patented by Fritz Klatte. Yet, PVC was commercialised 10 years before it was called a ”polymer” and 20 years before it was understood what polymer is.

The ability to create materials without depending on natural resources was quickly appreciated, and leading countries focused on search and production of novel plastics. During the World War II, nylon substitutes silk and organic glass, Plexiglas®, is used as a substitute for inorganic glass. Since the beginning of 20^{th} century the worldwide production of synthetic polymers continues to grow, and its role in the modern society is difficult to underestimate. So is the research in the field of polymer science: now, when the negative influence of plastics on the environment becomes more and more of a serious issue, researchers focus on development of novel environmentally friendly plastics. Furthermore, modern technology centred society still depends on many natural resources, that are sought to be replaced.

2.2 Classification

Since polymer materials cover a wide range of applications, they can be divided into categories based on several aspects. The aspects that play essential role in this work are monomer composition and material structure.

Monomer is an individual small molecule that corresponds to the repeating unit of a polymer. While repeating unit is a part of the polymer, in other words, it has at least one bond to neighbouring repeating units, monomers represent repeating units in the state of stand-alone molecules. Figure 2.1 shows graph representations of polypropylene along with its monomer propylene.

Polymers are produced by joining many monomers into a polymer chain through

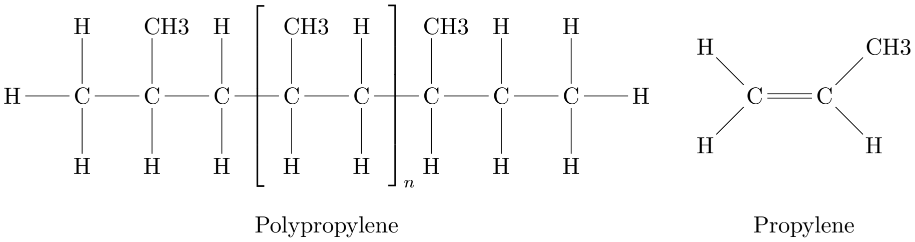


Figure 2.1: Graph representations of polypropylene along with its monomer propylene. Repeating unit of polypropylene is denoted by square brackets.

a chemical reaction known as polymerisation. The average number of repeating units per chain within material is called degree of polymerisation. Depending on how many repeating unit types are used to build the chain, polymers are divided into homopolymers and copolymers. The former are built with a single monomer, while the latter consist of two or more monomers.

It is worth noting that the chain structure, or the shape of the polymer chain, also affects material's properties [14]. In linear polymers, where each repeating unit is attached to not more than two others (Figure 2.2a), the chains are attracted through weak forces, like van der Waals' forces or dipole-dipole interactions. Branched polymers are similar to linear polymers in terms of interaction, but as the name implies, some of the repeating units may be connected to more than two others, forming branching structures (Figure 2.2b). These molecules exhibit lower packing efficiency, resulting in lower densities. Network polymers are branched interconnected polymeric systems (Figure 2.2c). Unlike branched polymers, where molecules remain discrete, in network polymers the chains are cross-linked, forming a macroscopic material. Because of permanent chemical bonds, such materials are more resistant to dissolution in solution or melt. The strongest polymer chain arrangement is when the polymeric chains are interconnected pairwise on a regular basis forming ladder-shaped two-strand structures (Figure 2.2d), unlike network polymers where connections are placed randomly within the bulk.

The research on polymer with special shape is limited and thus the information on chain structure is rarely present in databases. For this reason, the chain configuration

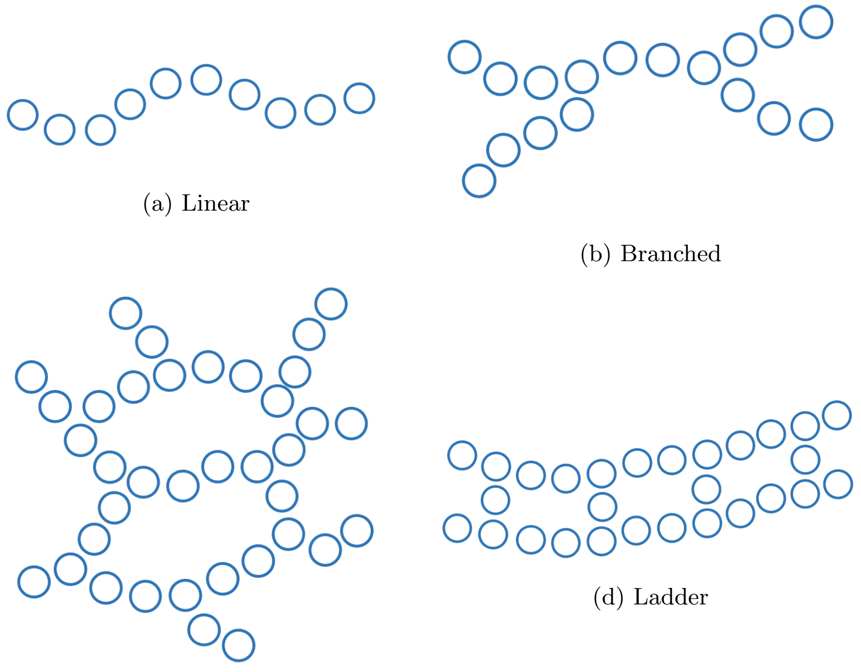


Figure 2.2: Illustration of different polymer chain structures: (a) linear chain, (b) branched chain, (c) network structure, (d) ladder structure.

could not be considered in the present study. It is nevertheless important to keep in mind that the chain structure remains another parameter that may lead to large deviation among measured property values for a homopolymer.

By macroscopic structure plastics can be divided into two groups: amorphous and crystalline. Amorphous polymers have no ordered structure within the bulk material; they are transparent and are usually brittle. In crystalline polymers, chains form ordered structures (crystals), and crystalline plastics are relatively strong and ductile and often opaque. Yet there exist no purely crystalline polymer. Instead, crystalline polymers are characterised by a degree of crystallinity, which is defined as fraction of crystalline regions within the material. Typically, polymers' degree of crystallinity lies in range $10\% \sim 80\%$, and the region between crystalline domains is filled with amorphous domain. Naturally, the degree of crystallinity affects most of the polymer properties.

Deeper notions on polymers, their properties and classification can be found in classic books on polymer physics, such as the one from van Krevelen [15], Askadskii [16] or Bicerano [17].

2.3 Thermal properties

Thermal properties are important in industry. They are crucial for all the stages of material life cycle: manufacturing, usage and recycling. Nonetheless, up until now there no rigorous theoretical model able to accurately predict them based on polymers' chemical structure. The reason is that a typical polymer chain consists of multiple thousands of atoms. These chains are entangled within material, making a complex structure often compared to a pasta bowl.

Furthermore, thermal properties are the result of both molecular interactions and dynamics. The main factors influencing polymer thermal properties are cohesive forces between molecules, topological and geometrical arrangement of atoms, chain stiffness and bond flexibility, and molecular weight (M_W) [17]. Polymer properties also largely depend on processing and experimental setup (such as cooling rate, *etc.*). These factors make the investigation of polymer thermal properties via first principles calculations extremely challenging [3].

Quantitatively, until now it was only possible, to roughly relate polymer thermal properties to other measured macroscopic properties, such van der Waals volume (V_W), or to use group contribution method [15], [16] (see Section 2.4.1). In the following subsection we introduce coefficient of thermal expansion, CTE, one of the principal thermal properties of polymers for which high accuracy prediction model is needed.

2.3.1 Thermal expansion

The coefficient of thermal expansion (CTE) is a property reflecting dimensional stability of a material under changing thermal conditions. There are several types of the coefficient: volumetric, area and linear. For practical reasons, the linear CTE is more often used. By definition, the coefficient of linear thermal expansion β is defined as follows:

$$\beta = \frac{1}{L(T)} \left(\frac{\partial \mathrm{L}}{\partial \mathrm{T}} \right)_p , \tag{2.1}$$

where L(T) is the length of the material at temperature T, ∂L is the change in length given ∂T difference in temperature at constant pressure p. The higher the CTE value is, the more a given material expands with increasing temperature. CTE is an industrially crucial property since mismatches in thermal expansivities between different materials lead to internal stress, and eventually to a failure, of a manufactured product.

Nevertheless, up until now there was no general theoretical or empirical model able to accurately predict the CTE for homopolymers based on their molecular structure. In case of ceramics or metals, which have well-defined rigid atomic structure, it is possible to estimate the CTE using first principles calculations [18], [19]. As we said earlier, such precise computations are unfeasible for amorphous polymers. Recently, it has been possible to evaluate the CTE for cross-linked epoxy polymers through accelerated ReaxFF, but the process relies considerably on human experience and yields in largely underestimated CTE values [20].

Empirical methods used until now relate the CTE of amorphous polymers to other measured macroscopic properties, such as glass transition temperature ($\mathrm{T_g}$) [21] or van der Waals volume ($\mathrm{V_W}$) [15], [16]. For some of crystalline polymers, the CTE can be evaluated through the morphology of crystals [22]. In case of copolymers or composite materials, the CTE is computed as a combination of individual components' CTE, and is not based on a whole chemical structure [23].

CTE depends on topological and geometrical arrangement of atoms, chain stiffness and bond flexibility [17].

2.4 Property prediction

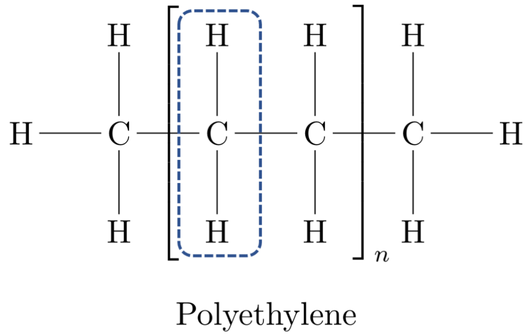
There exist three major semi-empirical methods for determination of polymer properties based on chemical structure that do not involve machine learning. They have been developed in a thirty years range starting from early 1970's and until the beginning of 2000's and remain unchanged since then.

2.4.1 Group contribution method

Group contribution method (GCM) is a semi-empirical method of estimation of molecular properties from chemical structure introduced by van Krevelen in 1970's [15]. He assumed that any given property for the whole molecule can be represented by a sum of partial contributions of its fragments, often referred as structural groups (such as individual atoms or groups of atoms, bonds, rings, *etc.*). These contributions are computed individually for every property and are based on available experimental data.

Estimation procedure always starts with the simplest compound. For polymers, it is polyethylene $(C2H4)_n$, which allows to compute the contribution for the most basic -CH2- group (see Figure 2.3). Next step will typically be to determine the contribution of the -CH-CH3- group by looking at polypropylene and using the contribution computed for -CH2- group. Then, contributions of other groups are computed step-by-step by looking at the experimental data of polymers of increasing complexity, dealing with one fragment at a time.

Clearly, such an approach is a rough estimation, since many properties are not additive in nature (for example, glass transition temperature). Also, every structural group may show different behaviours depending on its surrounding. Moreover, the method



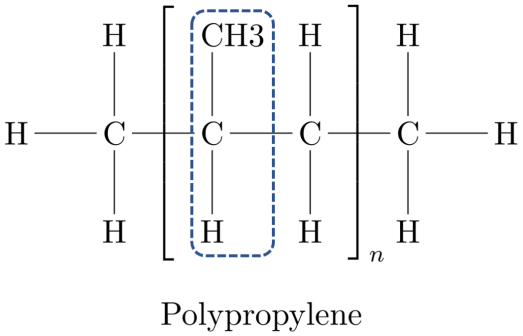


Figure 2.3: Graph representations of polyethylene (left) and polypropylene (right). Repeating units are denoted by square brackets. Functional groups whose contribution are being defined are denoted by dashed rounded squares.

is highly sensitive to the data used for weights computation. This sensitivity is unbalanced, since the first basic structural units impact the overall model the strongest. Also, contributions of individual fragments change depending on the material type and should be computed individually (for example, heat capacity for -CH3- group computed for polymers is three times larger than that of organic liquids [15], [24]). One of the strongest disadvantages of the method is that it is limited to the groups used for regression. In other words, it is impossible to evaluate properties of polymers containing a novel structure group of element.

Nonetheless, GCM has been used for polymer development, where quantum physics based precise computations are prohibitive in terms of memory and time, and experiments are often prohibitive in terms of budget. Until now GCM was one the most successful approaches in the field of polymer physics, showing reasonably good results for some molecular properties prediction [15]–[17].

2.4.2 Calibration based on standard polymers

Another semi-empirical method has been proposed by Askadskii [16], which was first published in Russian in 1982 and appeared on the international scene some 10 years later. As van Krevelen, Askadskii also uses regression, but instead of determining the contributions of fragments, does so for individual atoms. This solves the problem of applicability of the method for the development of polymers containing novel structural groups, which van Krevelen's method suffers from.

The reason why van Krevelen chose to focus on structural groups is that comparing to structural groups individual atoms are yet more sensitive to the chemical context. The precision of a model based on atom contributions based purely on experimental data would be compromised. To counteract this issue, Askadskii takes into account some physical principles which might be responsible for a given property value. For example, it is suggested that T_g is contributions of every atom are proportional to the part of the van-der-Waals volume that this atom takes within the total volume of repeating unit, and also takes into account intermolecular interactions (dipole-dipole, hydrogen bonds, *etc.*):

$$T_g = \frac{\sum\limits_i \Delta V_i}{\sum\limits_i a_i \Delta V_i + \sum\limits_j b_j}, \tag{2.2}$$

where a_i are partial coefficients of thermal expansion for i-th atom, ΔV_i is the van der Waals volume of the i-th atom and b_j are contributions of various types of intermolecular interactions. ΔV_i can be computed straight away by following geometry of particular configuration. Then, depending on the total number of parameters, $N_{param} = a_i + b_i$, their values can be found via regression by obtaining N_{param} experimental CTE values and solving a system of N_{param} equations:

$$\begin{cases} a_1\Delta V_{1,1} + a_2\Delta V_{1,2} + \ldots + a_n\Delta V_{1,n} + b_1 + b_2 + \ldots + b_j = \frac{1}{T_{g,1}} \left(\sum_i \Delta V_i \right)_1 , \\ a_1\Delta V_{2,1} + a_2\Delta V_{2,2} + \ldots + a_n\Delta V_{2,n} + b_1 + b_2 + \ldots + b_j = \frac{1}{T_{g,2}} \left(\sum_i \Delta V_i \right)_2 , \\ \ldots \\ a_1\Delta V_{m,1} + a_2\Delta V_{m,2} + \ldots + a_n\Delta V_{m,n} + b_1 + b_2 + \ldots + b_j = \frac{1}{T_{g,m}} \left(\sum_i \Delta V_i \right)_m . \end{cases}$$

However, the efficiency of the method for predicting properties of novel structures is doubtful. First, to achieve a better fit, Askadskii introduces some ad hoc corrections for numerous special cases (polyamides are treated independently, for polymers containing F or Cl atoms dipole-dipole interactions are neglected, *etc.*). Even though these corrections make sense from the physics point of view, such human intervention is an indicator of the lack of raw model's generalisability. Second, the parameters are found via solving a system of equations, where the right part of every equation corresponds to experimental value of the desired property. Since the number of parameters, N_{param}, is typically somewhere between 30 and 40, the overall model is entirely defined by 30 to 40 experimental values. Ultimately, the choice of polymers on which the model will be based requires field expertise and depends on human decision, which is again not enough to grant good generalisability.

2.4.3 Connectivity indices

The latest semi-empirical approach has been developed by Bicerano [17] in 1990's. Similar to Askadskii, Bicerano aims to get rid of structural groups introduced by van Krevelen, and to compute per-atom contributions. He treats molecular structure as graphs and computes so-called connectivity indices, which are related to the number of bonds per atom and atom valence. These 4 indices are calculated based on a predefined set of equations. Every property is assumed to be a linear combination of all or some of these indices, and the coefficients are found via regression.

Similar to Askadskii, Bicerano ends up introducing multiple corrections to achieve a better fit (such as replacing Si atoms with C, adding correction for cyanide or alamid groups, *etc.*). Some of these corrections lack theoretical basis, and the generalisation of the method is very questionable. Bicerano's approach is also limited to polymers containing C, H, O, N, F, Si, S, Cl, or Br atoms.

After all van Krevelen's group contribution method has become the most popular method among polymer community. It is simple and straightforward enough, and makes fewer assumptions comparing to other methods.

3. Background: Machine Learning in Materials Science

The present Chapter covers ML methods existing in the field of materials science. As stated in Chapter 2, the only prediction methods for polymer properties that exist at present are semi-empirical and predict properties based on polymer repeating unit chemical structure via regression. Similarly, ML also relies on existing experimental data and adjusts training parameters to fit the data. The advantage of ML is that it can cope with large amounts of data and is able to find complex patterns within the data without human intervention (while human is still responsible for providing relevant data). Therefore, there is much hope that ML can achieve similar or better results in predicting property values based on chemical structure.

3.1 Molecular representations

In order to develop a ML model based on chemical structure, it is important to choose molecular representation. Here we describe four molecular representations commonly used in the field of materials science.

Molecular descriptors

Molecular descriptors are the results of experimental measurements or theoretical calculations that are believed to be correlated with the property of interest. Some of the descriptors can be computed almost instantaneously by using the information about the structure (number of specific atoms, number of rings, molecular weight, *etc.*). Others rely on first-principles calculations to compute deeper level properties (highest unoccupied and lowest occupied molecular orbits (HOMO and LUMO), energy levels, ionisation potential, *etc.*). These computations do not require empirical parameters but may take significant amount of time. The third type of descriptors relies on experimental values. They can be used when there exist a property highly correlated to the property of interest, whose experimental measurement is significantly easier to perform. Successful choice of a suitable set of molecular descriptors is far from being trivial and requires extensive expertise in the field related to the property of interest [25].

Fingerprints

A fingerprint represents binary hashed information about every atom within a given molecule. It is a compressed version of molecular descriptor that is developed specifically for computer usage. Fingerprints are commonly used together with diverse regression techniques or random forests (RFs). However, similar to molecular descriptors, the choice of fingerprint demands significant domain knowledge and often involves trial and error [8].

Molecular graphs

Molecular graph is a 2D molecular representation depicting atom placement for a given molecule unfolded flat. While molecular graphs do not provide information on how the molecule looks like in 3D space, they do represent molecular structure entirely, in other

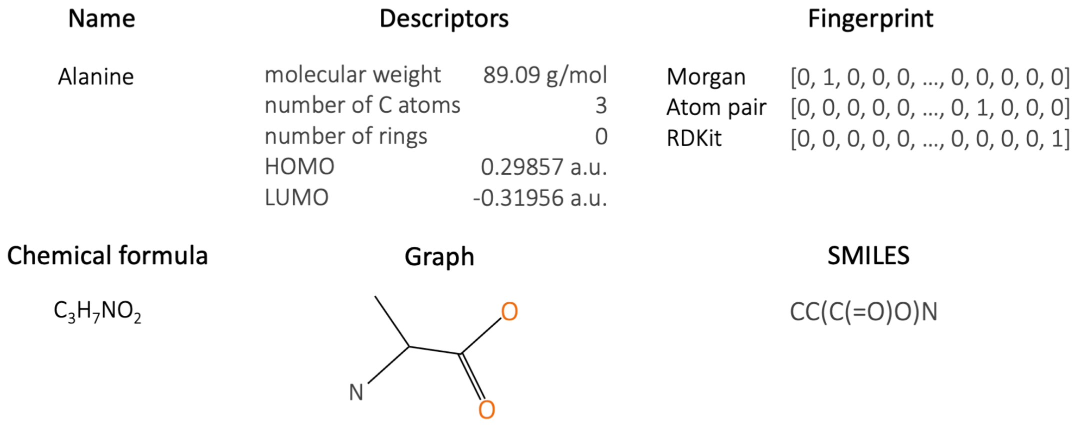


FIGURE 3.1: Various molecular representations for alanine molecule.

words, allow to distinguish molecules.

SMILES

While graphs are very convenient for characterising molecular structure, they are not easy to store in databases. In order to make it possible, SMILES (simplified molecular input line entry system) format has been developed [26]. SMILES string is essentially a text representation of molecular structure derived from a molecular graph. Since the derivation is done by following a fixed set of rules, SMILES are straightforward to create and do not require any primary knowledge on the data. SMILES notation represents the relative position of atoms within a two-dimensional space and distinguishes bond types, aromaticity, isotopes and permits specification of stereoisomers.

The four described representation are visualised in Figure 3.1.

3.2 Applicable machine learning methods

The choice of the machine learning method naturally depends on the input format. Molecular descriptors and fingerprints are typically used with various sorts of regressions

(Gaussian process regression [5], support vector regression [6]), RF and other quantitative structure–property relationship models [7]. The combination of fingerprints with RF is often found to be the most accurate over a large range of applications [27]–[30].

As for the molecular graphs, they can either be treated as images or as graphs, with individual atoms as vertices and bonds as edges. Consequently, the image format can be combined with convolutional NNs (CNN), whereas graph representations can be fed into graph NNs (GNN).

Finally, SMILES representation can be treated with natural language processing (NLP) techniques.

Below we briefly introduce the most prominent ML methods, showing state-of-the-art in materials science.

Random forest

RF is a widely known machine learning algorithm based on the combination of decision trees [31]. Its versatility and robustness with respect to noise make it an excellent predictor for diverse tasks in various fields. From another hand, the nature of decision tree prohibits extrapolation, in other words RFs cannot predict the property values out of the range of the values used for training.

Graph neural networks

Most recently, graph NNs (GNNs) become one of the dominant ML methods [32], [33]. Thanks to versatile graph format, GNNs set state-of-the-art performances in many research fields from antibiotics discovery [34] to traffic speed prediction [35]. One of the most prominent GNN types are GNN with attention [36], or graph attention networks [37]. They allow to pass the information between neighbouring nodes during the training in a weighted manner, which allows for better generalisation as well as for visualisation of the prediction results.

Graph format may also be considered as the most suitable format for direct molecular structure description. For this reason, in the last couple of years GNNs become increasingly popular for building ML predictive models within such fields as biology, chemistry and materials science [38]–[40]. Yet, GNNs are known to suffer from long-range dependency problem [41]. While for some specific tasks this may not be a crucial obstacle, for most of the problems long-range interactions play an important role. Specifically, thermal properties are largely defined by inter-molecular interactions and the shape of the molecule. Therefore, GNNs might not be the optimal type of neural architecture for prediction polymer thermal properties.

Natural language processing methods

NLP is a machine learning field that is focused on text data. Therefore, it deals with long-term dependency problem within data: semantically related words can be placed far away one from another within a phrase. This situation is similar to components of SMILES string. Some of the atoms that do not bond directly one to another may still interact within the physical space. Also, due to the SMILES encoding rules, some of the atoms that share a direct bond can still be quite separated within a SMILES string. This makes weak interactions, *i.e.* interactions that do not form actual chemical bonds, to be similar to strong bonds in terms of SMILES encoding. In other words, a model that learns from SMILES strings might have good chances to discover weak interactions present in a molecule.

One of the key methods of NLP that solves long-term dependency problem is long short-term memory units, or LSTMs. LSTMs are neural cells that process input data string from the beginning to the end and compresses the information about every semantic component that it meets on the pass through so-called gating mechanism. Semantic components can be a letter in a word, a word in a phrase, or an atom or a bond within a SMILES. When passing through gates, the weights used for the gates determine whether to keep or to remember given information. Therefore, LSTM is able to keep in memory encoded information on already seen parts of the string, and to find relations within it. The weights used for the gates are trainable and are learnt during the overall model training.

Another great technique that has emerged from NLP is the attention mechanism [36]. Attention is an elegant and computationally cheap way to look at an input string as a whole, and to pick its most influential parts towards the final prediction. Not only attention helps to grasp the overall relationships within the input string [42], but also allows to implement visual interpretation of the model, by looking at the weights of each of units comprising a string.

Prediction of physicochemical properties based on the structure is not the only aim of a predictive model. Another role of predictive models, and probably the most important one, is to solve a reversed problem of structure generation based on the desired value of the property of interest. It seems that language based models show better results on the tasks of molecule generation compared to GNNs [43].

Finally, LSTM-based models are more compact than GNN ones, which is suitable for small data (less risks of overfitting). Therefore, for the purpose of this thesis, it is reasonable to consider SMILES as phrases written in chemical language, and to apply NLP methods to translate them into the values of properties of interest.

To summarise, molecular descriptors and fingerprints require domain knowledge and are specific to the application. Molecular graphs are easy to produce, but GNNs tend to perform poorly on tasks with long-range interactions [41]. Since the LSTM-based neural networks (NNs) are explicitly designed to avoid the long-term dependency problem and SMILES are easy to produce. Theoretically speaking SMILES should allow to grasp the same kind of information that can be computed via first principles calculations, because first principle calculations compute property values based on the chemical structure of a molecule. In contrast, the ML models based on structure do not use hard-encoded quantum physics rules in order to deduce a molecule's properties. Instead, these laws emerge from the data. This means that the quality of the prediction strongly depends on the dataset: its quality, diversity and size.

Based on the above, in the present research we chose the SMILES-X software [P1] for building predictive models. We also use a RF model with fingerprint inputs as a baseline proxy.

3.2.1 SMILES-X

The SMILES-X is a software package designed specifically for the prediction of physicochemical properties of molecules [P1]. It takes SMILES as input and features LSTM-based architecture which exploits the latest advances from NLP field. NLP is focused on processing information in text format. For example, NLP algorithms can automatically rate a book based on its review. Similarly, SMILES-X treats SMILES as phrases written in chemical language and translates them into values of properties of interest.

Pipeline and architecture

The overall SMILES-X pipeline is as follows.

Augmentation First, the SMILES-X implements automatic data augmentation. While the order of composing a SMILES string from a graph is fixed, it can be written starting from a different atoms. Therefore, for a graph consisting of N_{atom} atoms there can exist up to N_{atom} individual SMILES representation. This situation is similar to images, where rotation does not cause the object on the image to change. In SMILES-X this input invariance is compensated by data augmentation, so that for every compound the extensive list of possible individual SMILES is created. This allows the model to deepen its *understanding* of structure-property relationships, by becoming agnostic to the SMILES multiple arrangements.

Tokenisation Then, the network hyperparameters are fixed in accordance with the data. Prior to entering the neural architecture, the original SMILES is broken down into tokens. This is a standard approach in NLP, where the text is broken into semantic units. In case of SMILES, such units may be individual atoms, bonds, branches and other auxiliary symbols used for SMILES encoding, such as ring numbers (*e.g.*, '[N]', '[Cl]', 'c', '=', '$', '3'). In addition, in case of the polymers there exist an extra token indicating points of attachment between repeating units. It is represented with the wildcard asterisk symbol ('*'). All tokens are buffered with whitespace tokens so that all the SMILES within the dataset has the same length. Finally, whitespaces are also added in the beginning and the end of each string. Beginning and termination tokens are the same, since SMILES are direction agnostic.

Bayesian optimisation In the original SMILES-X software both geometry related and trainable hyperparameters are optimised via Bayesian optimisation (BO). The method is based on the iterative evaluation of various combination of hyperparameters, θ, by training the corresponding neural architecture. The score is assigned based on the best validation root-mean-square error (RMSE) achieved during training.

Specifically, BO is used to optimise the black-box function $f_{\mathrm{RMSE}}(\theta)$ to ultimately predict its minimum. In reality, the surrogate Gaussian process function is estimated instead. Technically, the process is implemented in a two-phase manner. During the first, initialisation phase, every next hyperparameters combination θ is selected at random. This allows to cover wide space and minimise the risks of being stuck at a local minimum. Then, during the optimisation phase the algorithm focuses on finding the minimum of the achieved function (exploitation) while simultaneously maximising the information gain over the function shape (exploration). The trade-off between exploitation and exploration is defined by the user via a parameter within the so-called acquisition function, which is used to select the most promising θ to be tested. In SMILES-X, BO is implemented via GPyOpt[44] Python package, which uses 'expected improvement' acquisition function.

By default, the SMILES-X uses 20 epochs during BO process, but this parameter can be easily changed. Typically some 20 combinations are evaluated at random during the first phase, followed by 30 evaluations during the optimisation phase.

After the data is ready and the optimal set of hyperparameters is determined, the training phase begins. it is time to train the data. The SMILES-X neural architecture is based on the {Embed, Encode, Attend, Predict} scheme. This framework was conceptualised by Matthew Honnibal in 2016 [45] and proved state-of-the-art in NLP. Another important point is that the overall architecture is quite shallow, which should be suitable for small data so typical in materials science. The overall neural skeleton is given on Figure 3.2. It consists of the following 4 steps.

Embedding The role of embedding is to represent each component of a SMILES by a float vector of arbitrary length. For every dataset, a vocabulary of all the semantic units used within a given dataset (SMILES components). This representation is randomly initialised and trained as a part of the whole network.

Encoding Encoding is done via bidirectional LSTM followed by a dense layer. Since SMILES can be read from start to end and from end to start while representing the same molecular structure. The key feature of LSTM unit based NNs is their ability to grasp distant relations between the features in the input. In NLP it has been developed to deal with distant words within text that are related one to another. In SMILES-X it is important to take into account interactions between atoms that are placed far apart within a SMILES but are situated near each other in physical space. LSTM cells store information about every token while sequentially passing

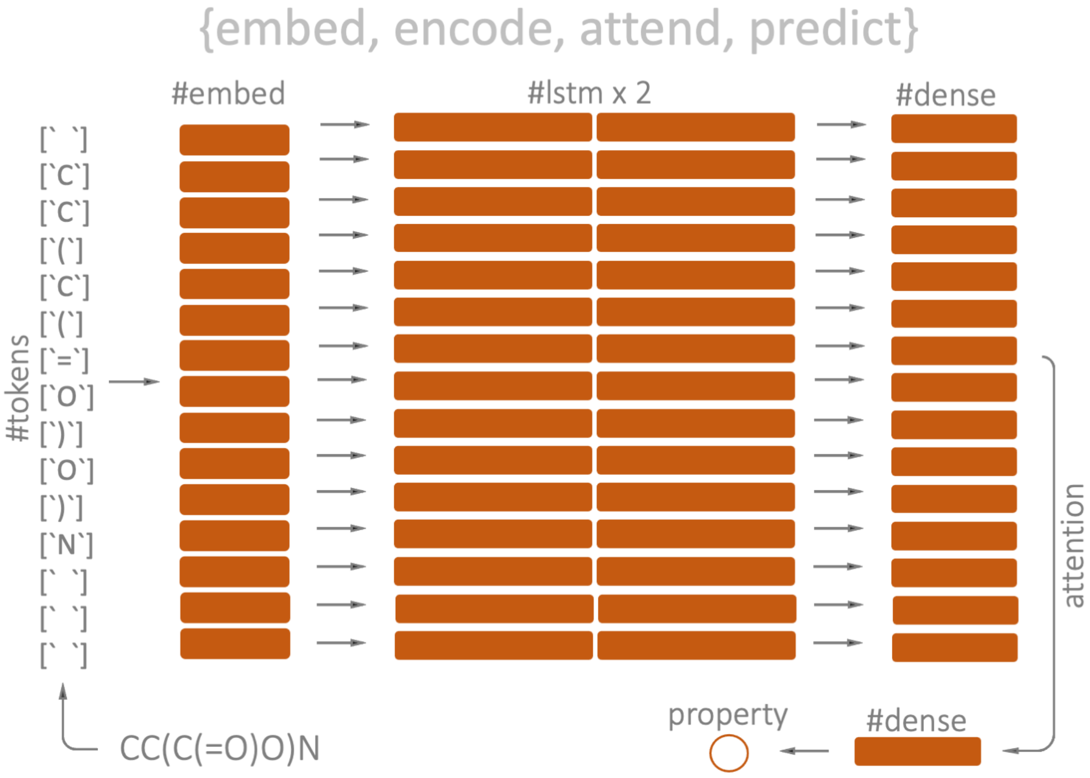


Figure 3.2: The overall skeleton of the SMILES-X neural architecture.

through SMILES. Therefore, the output of encoding layer holds information on forward and backward passes concatenated together.

Attention Attention layer plays dual roles. Its primary role is to enhance LSTM by looking at the whole SMILES simultaneously and assign individual weights for each token [42]. In SMILES-X, so-called soft attention is implemented. What the attention layer practically does is to compress the tensor $\mathrm{H} \in \mathbb{R}^{\mathrm{n_{tokens}} \times \mathrm{n_{dense}}}$, yield by encoding, into an $\mathrm{n_{dense}}$ vector c with minimum information loss:

$$
\begin{aligned}
e &= tanh(\mathrm{H} \cdot \mathrm{W_a} + \mathrm{b_a})\,, \\
\alpha &= \frac{\exp(e)}{\sum_{i=1}^{\mathrm{n_{tokens}}} \exp(e_i)}\,, \\
c &= \mathrm{H}^{\mathrm{T}} \cdot \alpha\,,
\end{aligned}
\tag{3.1}
$$

It is graphically represented on Figure 3.3. The secondary role of the attention is to provide for visual interpretation. The attention mask α contains weights for every token within a given SMILES, *i.e.* token's importance towards the final prediction. More on the interpretation and examples are given in the Section 5.6.2.

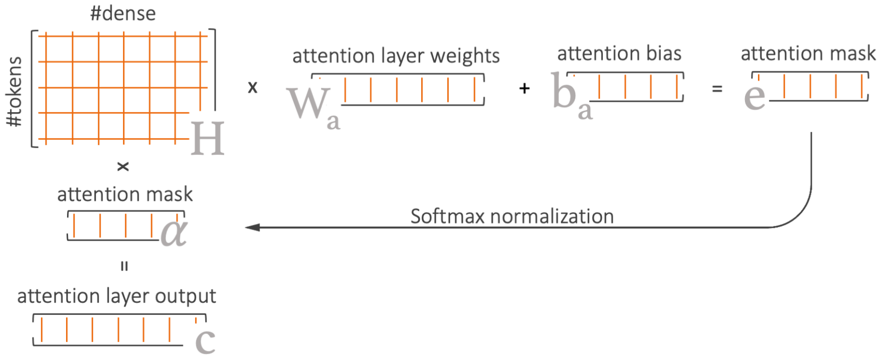


FIGURE 3.3: Illustration of the soft attention mechanism implemented within the SMILES-X.

Prediction Prediction layer is a simple fully-connected layer mapping the $\mathrm{n_{dense}}$ vector output by attention into the predicted property value.

When applied to three benchmark physical chemistry datasets issued from the MoleculeNet[46], SMILES-X showed the state-of-the-art performance. Notably, its results are comparable to the graph attention networks published later [39]. The details on the SMILES-X software can be found in the corresponding paper [P1].

4. Deep Learning: Neural Architecture Search (NAS)

While the overall neural architecture of the SMILES-X is fixed, we still need to define the number of units in the embedding, LSTM and dense layers. Finding the optimal neural architecture is one of the cornerstone steps in deep learning. Commonly, there is an extremely large number of parameters to be tuned, and the process can be overwhelmingly slow. When deep networks were starting to emerge, the architecture shape was chosen by hand following trial and error. Gradually, neural architecture search (NAS) field has emerged as a way to automatise and accelerate the decision taking, shifting the task from humans to machines. Eventually NAS has become one of the most popular topics among the deep learning community.

4.1 Conventional methods

The first attempts to find the most suitable network structure were done through evolutionary algorithms [47]–[50]. There, several architectures are mutated in various ways (e.g. adding or removing a layer, changing activation function, etc.), and the resulting offsprings are evaluated through training. The best performing of the offsprings are added to the population for the next step, and the procedure is repeated for a given number of steps. This method has been used since back in the 1990's [51] and shows one of the best performances until now [52].

Similarly, Bayesian optimisation [53] is used to optimise a black-box function of geometry parameters by probing its values via training for a subset of architectures [54]. This method has shown a few state-of-the-art performances in the period between 2013 and 2020 [55]–[57] and is implemented in the original SMILES-X package for the simultaneous optimisation of the geometry with batch size (BS) and learning rate (RT) [P1].

In 2016 Zoch et al. [58] proposed the use of reinforcement learning [59] to build neural architectures from scratch. There, a so-called controller NN is trained to build a child-network — the network to be used for the final training and prediction. The original method demands tremendous amount of child-model training and is extremely lengthy. Several related works show significant acceleration of the process by reducing the search space [60] or introducing weight sharing [61], known as one-shot NAS.

An extensive overview of the NAS methods has been recently done by Thomas Elsken et al. [62].

Evaluating geometries through training clearly brings multiple disadvantages. The most obvious of them is that training is a computationally expensive process, and large-scale geometry evaluation often can not be carried on massive datasets. For the same reason architectures are usually trained with a single random seed, therefore, the statistical validity of the selected architecture is questionable. Moreover, architectures are typically compared with the same set of training hyperparameters, which might not be optimal for each of them. Such method yields an architecture optimal for the chosen hyperparameters, but likely not the overall optimal architecture for the task. Training also implies the usage of hand-labelled data, which brings in some human error – ImageNet dataset, for instance, has a label error of about 6% [63]. Finally, none of the above NAS methods answer to the question why do some geometries perform better than others.

4.2 Towards trainless NAS

As a step towards trainless NAS, in 2018 Istrate et al. [64] have introduced a small LSTM-based model, that allows to predict architecture's performance without training it on the data of interest. Their model predicts an architecture's potential for a given data complexity. The data is taken from a so-called lifelong database of experiments. While it is possible to accelerate geometry search for some closely related problems (existing data belong to image recognition field), this approach cannot be applied widely. The straightaway restriction of this method is that there should already exist some data of a similar complexity within the database, and the available networks are limited to already existing ones (focused on image classification). Moreover, with time the overall procedure might lead to a bias, *i.e.*, a most often predicted architecture in the beginning will have yet more chance to be output in future, thus ”locking” it at the top position.

A similar approach is proposed by Deng et al. [65]. The authors encode layers composing the network into vectors, and bring them together with a predictive LSTM layer to build numerical representation of a network. A multilayer perceptron model is trained to predict the architecture with the highest prediction accuracy. Therefore, in order to use this method one needs to first train a set of architectures to acquire their trained accuracies, and then to train the predictive model on top. Moreover, since the final decision is made by a neural model, this method does not provide a reason why a given architecture has been chosen.

The first work that investigates fundamental architectural properties of NNs in order to attain fully trainless NAS is proposed by Mellor et al. [66] in 2020. The authors introduce a NASWOT metric (acronym for ”neural architecture search without training”) by looking at how networks distinguish the inputs based on signal propagation through rectified linear unit (ReLU) activation functions during a single forward pass. While showing relatively good performance, their scoring metric is restricted to ReLU based architectures. Using the NAS-Bench-201 benchmark database [67], the authors show that their metric is able to distinguish one of the best neural architectures among many with consistent success. To the best of our knowledge, this is the only approach that aims to give an explanation of NN's performance based on its structure.

On another end, there are a few papers, indicating that the best trained neural architecture often shows a better untrained accuracy. For example, the work of the UBER team [68] mentions that the best final architecture shows nearly 40% accuracy on MNIST dataset [69] at initialisation. David Ha and Adam Gaier [70] have presented a NAS algorithm which builds an architecture based on the untrained score. Their score is taking into account both the number of parameters contained within a model, which they seek to minimise, and the mean accuracy, which is being maximised. The mean accuracy is computed over several initialisations of the child model using a set of constant weights (single value for all the weights). They report that the resulting model achieves 82.0% ± 18.7% on MNIST data with random weights at initialisation, and over 90% when the weights are fixed to the best performing constant ones.

These findings imply that NNs might have an intrinsic property, which defines their prediction performance prior to training. Such property should not depend on the values of trainable parameters (weights), but only on network's topology.

4.3 Scoring metric search

Our work on trainless NAS can be divided into three parts. In the first part, Section 4.4, we conduct an extensive MNIST study to explore dependencies between various untrained statistics and the trained accuracy. For this, we train a range of fully-connected NNs on a reduced MNIST data, with multiple seeds and learning rates. In the second part, Section 4.5, the most promising statistical property is tested on existing large-scale benchmark datasets, featuring more complex neural geometries and data. Here we confirm generality of the selected property as a scoring metric for fully trainless NAS discuss its weak points and outline the ways to improve its performance.

4.4 Creating statistical NAS benchmark dataset with MNIST

The first attempt to reach trainless NAS is to verify how untrained metrics relate to the trained performance of a model. It is well known that random weight initialisation of a NN might have great influence on its outputs. In order to cancel out the influence of the weights and to bring out the architectural component, we needed to perform multiple weights initialisations to assess averaged networks' performances. Therefore, we created a dataset of trained NNs with multiple random initialisations.

The SMILES-X architecture was be suitable for such task, as we did not have enough of computational resources to perform extensive tests with multiple seeds. In addition, there exist no reliable molecular dataset that would be large, clean and easy-to-fit. Furthermore, as the SMILES-X structure is quite specific, it would be difficult to generalise our findings, share them with the ML community or make definitive conclusions on the subject.

For this, we have opted to perform tests on the well-known and simple MNIST dataset [69] by fitting it with shallow fully-connected NN. We have trained every neural architecture 100 times with random seeds. This granted us access to the mean and average accuracies before and after training, and allowed to analyse relationships of obtained metrics.

Moreover, every architecture may respond differently training hyperparameters (BS, LT), and therefore might require different hyperparameters to achieve its best performance. To take this factor into account, we have trained each architecture with several

LTs. As for BS, many of trained architectures showed the same optimal BS, so its value has been eventually fixed.

4.4.1 Search space

To keep things as simple as possible, we limit the search space to fully-connected NN composed of 2 hidden layers. The number of units in each hidden layer is set to be one of the 12 values in $[8, 16, 24, 32, 56, 64, 96, 128, 256, 512, 1024, 2048]$, making a total of 144 possible architectures. More complex architectures would inevitably bring in more uncertainty related to initialisation, activation functions, etc. Also, with larger architectures it would not be possible to perform training with multiple seeds. Lastly, since the MNIST dataset is quite easy to fit, most of these vanilla networks are already able to fit it with good accuracy ($\sim 93\%$).

4.4.2 Reduced MNIST dataset

MNIST is a classic benchmark dataset in the field of deep learning [69]. It consists of handwritten digits from 0 to 9. However, early tests revealed that it is too simple to see the difference between different architecture performances: most of them achieved similar accuracy after training. That is why we decided to slightly increase problem complexity by reducing the data available for training. We reduce the size of the training set, leaving 20 data points per class (200 data point in total). Not only it helps to better distinguish architecture performance upon training, but also this trick significantly accelerates the training process. On the same time, both the validation and test sets are entirely preserved, containing 5000 data points each. No data augmentation is applied.

4.4.3 Training scheme

Every NN is initialised and trained with 100 different seeds between 0 and 99, and 6 learning rates $[0.0001, 0.0003, 0.001, 0.003, 0.01, 0.03]$ (600 trainings per architecture, 86400 trainings overall). The batch size N_{batch} is fixed to 50, which we found showing the best results for a wide range of architectures within the search space. The models are built with Keras [71] and Tensorflow [72] and trained for 200 epochs using 3 NVIDIA Titan V GPUs. Weights are initialised using the He uniform initialiser [73], which is used together with ReLU activation function [74] for hidden layers and Adam optimiser [75] with default decay rates (0.9 and 0.99 for the first and second moments, respectively).

The final weights are based on the epoch with the best validation accuracy after a burn-in period of 50 epochs. Ignoring the first quarter of the training process is based on experience, since the validation loss of small noisy data tends to demonstrate random behaviour in the beginning of the training, leading to faulty results.

For each architecture only the learning rate showing the highest average training accuracy. This is done to insure that neural architectures are compared in a fair way, each showing its best performance. Afterwards, mean untrained error μ_u^{acc}, mean trained error μ_t, together with their respective standard deviations (σ_u^{acc}, σ_t) are calculated.

The pseudocode for the MNIST [69] training process is given in Algorithm 1.

4.4.4 Results of the MNIST study: σ_u^{acc}

The existing machine learning literature suggests that the best trained architecture may also show high untrained performance [68], [70]. We expect, thus, to see some tendency between mean accuracies prior to and after the training. We denote these accuracies as μ_u^{acc} and μ_t, respectively. Against our expectations, there was no clear correlation between these two metrics, as shown in Figure 4.1a. Instead, surprisingly, the mean trained accuracy μ_t showed some relation to the untrained standard deviation σ_u^{acc}: even though there is no linear correlation, the lowest σ_u^{acc} values belong to architectures from the top performance range (Figure 4.1b).

We have also observed that lower means μ_u^{acc} correspond to lower standard deviations σ_u^{acc} (Figure 4.2). Indeed, lower accuracy values lead to proportionally lower mean and

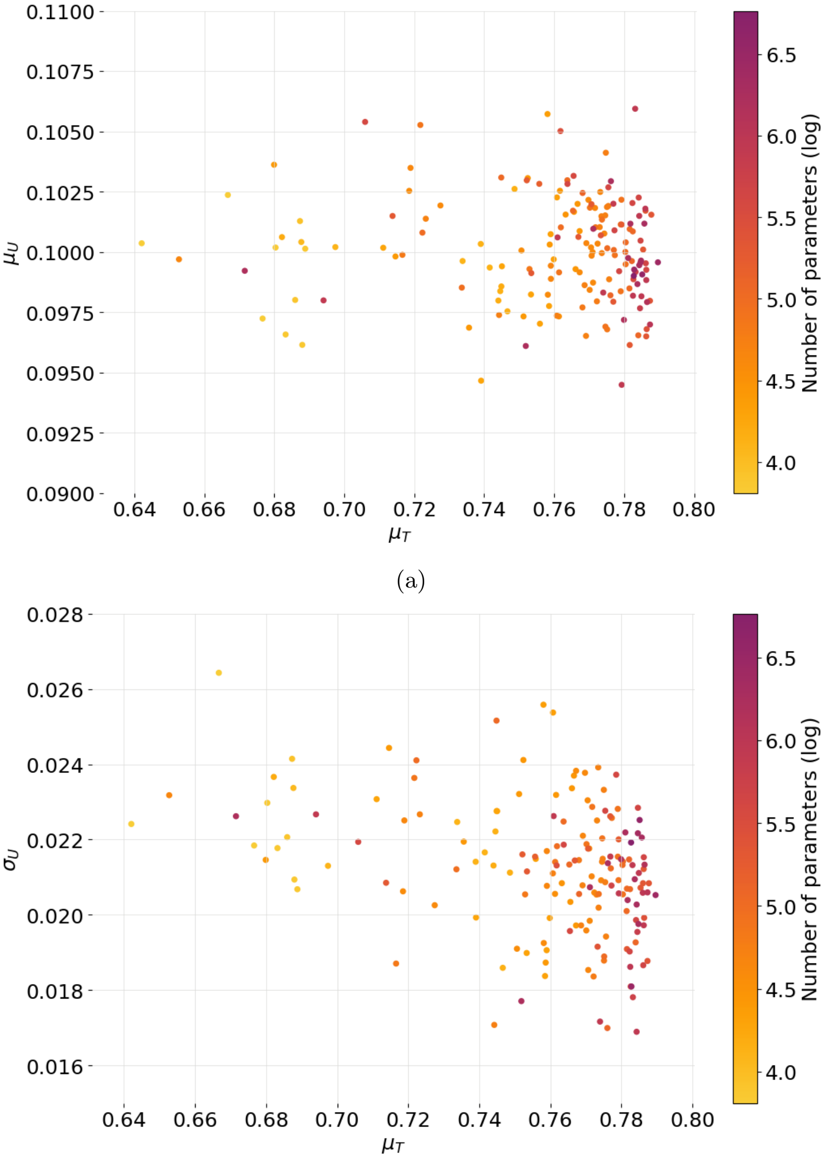
Algorithm 1 MNIST training pseudocode

```
Load the data
Split data on train/val/test sets
for cat in categories do                                          ▷ creating reduced train set
    Randomly pick 20 points from the original train set
end for
for nunits_layer_1 in $[8, 16, \ldots, 2048]$ do
    for nunits_layer_2 in $[8, 16, \ldots, 2048]$ do
        for lr in $[0.0001, 0.0003, 0.001, 0.003, 0.01, 0.03]$ do
            for seed in range($N_{init}$) do
                Build initial model
                Assess the untrained accuracy $U_i$                      ▷ untrained accuracy
                Train the model
                Select the final weights based on the best validation accuracy
                Compute test set accuracy $T_i$                          ▷ trained accuracy
            end for
            Compute means and standard deviations

                $\mu_t = \frac{1}{N} \sum_{i=1}^{N_{init}} T_i, \quad \sigma_t = \sqrt{\frac{\sum_{i=1}^{N_{init}} (T_i - \mu_t)^2}{N_{init}}}$

                $\mu_u^{acc} = \frac{1}{N} \sum_{i=1}^{N_{init}} U_i, \quad \sigma_u^{acc} = \sqrt{\frac{\sum_{i=1}^{N_{init}} (U_i - \mu_u^{acc})^2}{N_{init}}}$

        end for
        Select the best performing learning rate based on $max(\mu_t)$
                                  ▷ one set of [$\mu_t$, $\sigma_t$, $\mu_u^{acc}$, $\sigma_u^{acc}$] per architecture
    end for
end for
```



(b)

FIGURE 4.1: Mean untrained accuracy μ_u^{acc} (a) and standard deviation of untrained accuracy σ_u^{acc} (b) against mean trained accuracy μ_t, all three computed over $N_{init} = 100$ initialisations. One point stands for one architecture. The colours represent the logarithm of the number of parameters for a given architecture.

standard deviation. Therefore, minimising standard deviation alone may bias towards the networks that show overall low untrained accuracies U_i. To compensate for this effect, we normalise the standard deviation σ_u^{acc} by the mean μ_u^{acc}:

$$CV_u^{acc} = \frac{\sigma_u^{acc}}{\mu_u^{acc}}.$$

The resulting parameter CV_u^{acc} is known in statistics as the coefficient of variation, or relative standard deviation. When plotting CV_u^{acc} against trained accuracy μ_t in Figure 4.3, tendency becomes yet more clear: selecting the architectures with low CV_u^{acc} leads to high trained accuracy μ_t.

When looking for a NAS scoring metric, it is reasonable to consider how it correlates with the number of parameters contained within the network. It has been shown earlier that bigger does not necessarily mean better [76], [77]. Even though there is a higher chance for a bigger network to contain a subnetwork, capable of successfully fitting the data [78], there is also an increasing risk of overfitting, and increasing training time. We can confirm the effect of the performance saturation with our toy MNIST model both for the totality of parameters, and for the parameters in a single layer, as demonstrated in Figure 4.4.

Therefore, in order to find optimal architecture regardless the number of parameters, one should use a scoring metric uncorrelated with them. Figure 4.5 shows that there is no significant correlation between CV_u^{acc} and the number of parameters.

Taking all the above into consideration, in this part of the study we conclude that $CV_u^{acc} = {}^{\sigma_u^{acc}}/_{\mu_u^{acc}}$ might be a suitable trainless scoring metric for NAS.

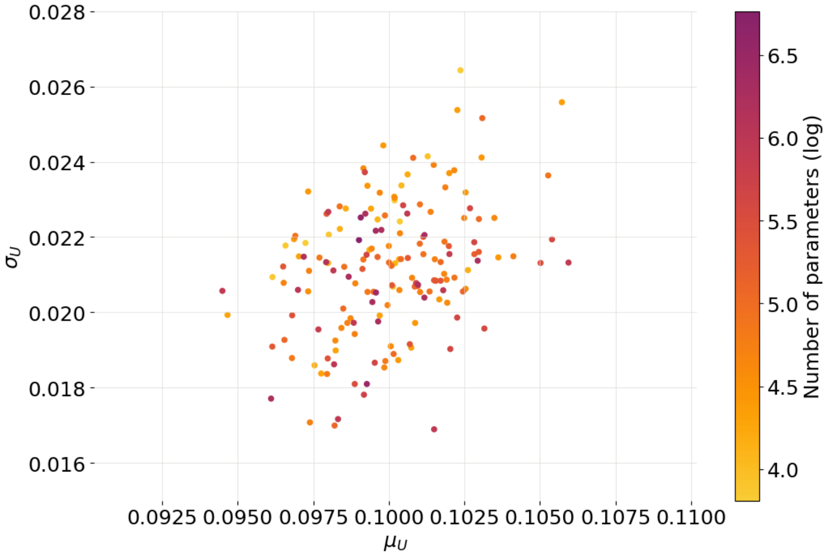


FIGURE 4.2: Mean untrained accuracy μ_u^{acc} against standard deviation of untrained accuracy σ_u^{acc}, computed over $N_{init} = 100$ initialisations. One point stands for one architecture. The colours represent the logarithm of the number of parameters for a given architecture.

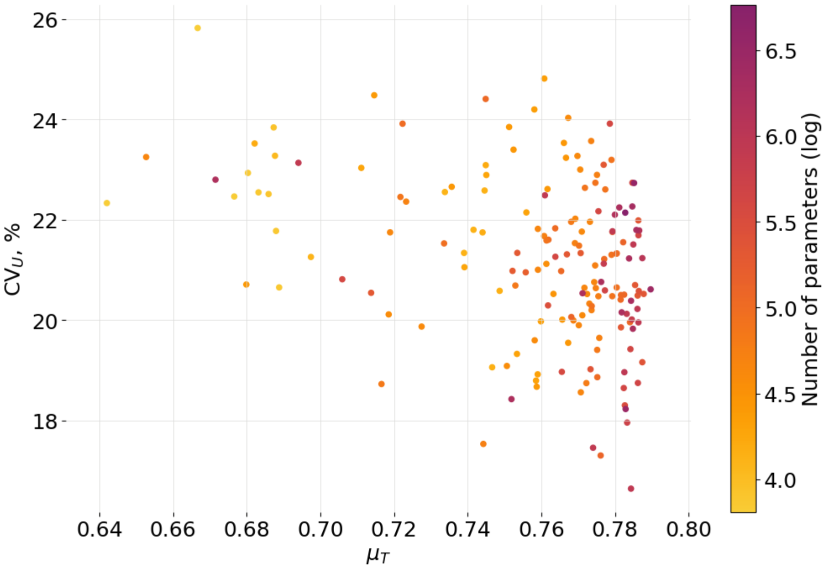
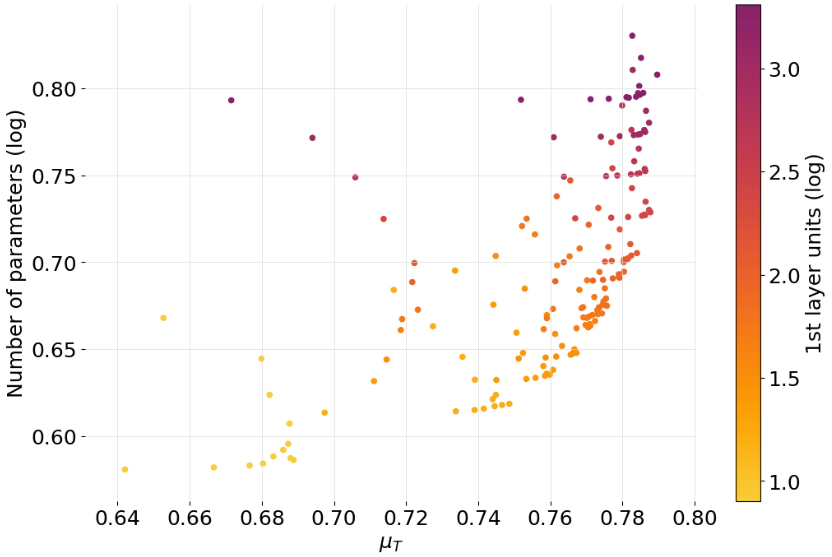
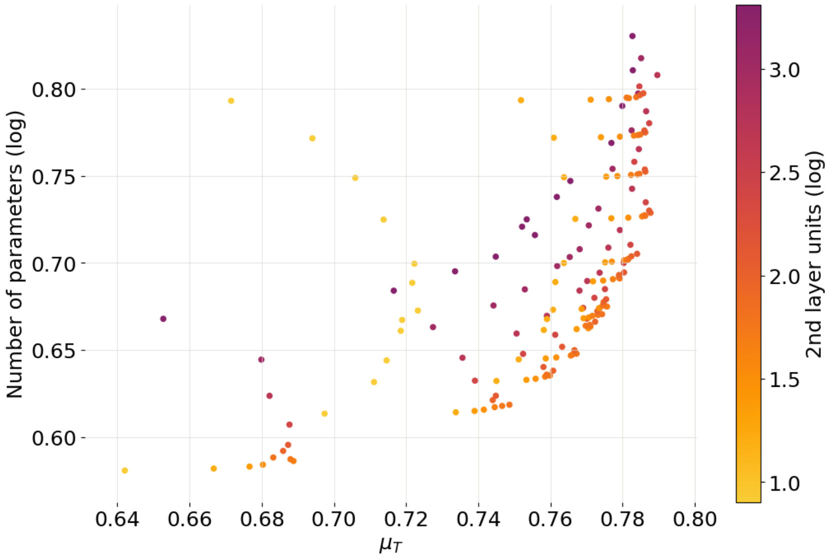


FIGURE 4.3: Coefficient of variation of the untrained accuracy CV_u^{acc} (%) against mean trained accuracy μ_t, both computed over $N_{init} = 100$ initialisations. One point stands for one architecture. The colours represent the number of parameters contained within an architecture.



(a)



(b)

FIGURE 4.4: Number of parameters against the architectures mean trained performance μ_t, computed over $N_{init} = 100$ initialisations. One point represents one architecture. Colours represent the number of units in the first layer (a) and the second layer (b).

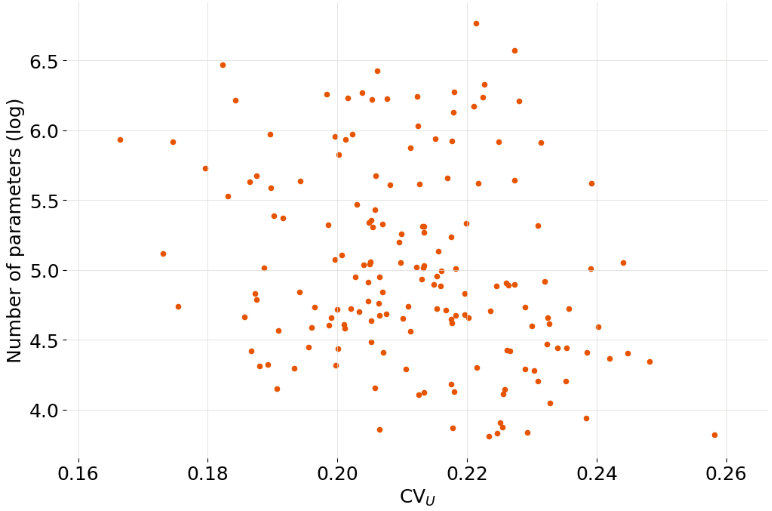


Figure 4.5: The number of parameters against the scoring metric CV_u^{acc} (%), or the coefficient of variation of the untrained performance, computed over $N_{init} = 100$ initialisations. One point stands for one architecture.

4.5 Validation of the CV_u^{acc} scoring metric

4.5.1 NAS-Bench-201 benchmark

To validate the generalisation power of the CV_u^{acc} metric it is necessary to apply it to more complex kind of NNs and different datasets. For this, we selected the NAS-Bench-201 dataset of fully trained residual neural architectures. This is a set of architectures with a fixed skeleton, consisting of convolution layer and three stacks of cells, connected by a residual block. Each cell is a densely-connected directed acyclic graph with 4 nodes, 5 possible operations and no limits on the number of edges, providing a total of $15,625$ possible architectures.

Datasets

Each of the architectures from NAS-Bench-201 [67] is trained on three major datasets: CIFAR-10, CIFAR-100 [79] and ImageNet [80]. Since the original CIFAR datasets do not contain a validation set, the NAS-Bench-201 authors created one by splitting the

Table 4.1: A summary over the datasets used in this thesis: number of classes, image resolution and splitting schemes (in thousands) for reduced MNIST, CIFAR-10, CIFAR-100 and ImageNet16-120.

Dataset	Classes	Resolution	Train/val/test (K)
Reduced MNIST	10	28x28	0.2/5/5
CIFAR-10	10	32x32x3	25/25/10
CIFAR-100	100	32x32x3	50/5/5
ImageNet16-120	120	16x16x3	151.7/3/3

original data. In case of CIFAR-10, the training set is split into halves to form the validation set, leaving the test set unchanged; for CIFAR-100, the test set is split in halves to form the validation set and the new test set. For the sake of computational tractability, a simplified version of ImageNet is used [80]. All the images are downscaled to 16x16 pixels, with 120 classes kept, forming a new ImageNet16-120 dataset. Data augmentation is used for all datasets; augmentation schemes differ slightly between CIFAR [79] and ImageNet [80] due to the difference between input image sizes.

An overview on all the data used for the trainless NAS metric development in this thesis is given in Table 4.1.

Training

The training is done using up to 3 different seeds (a single seed for the majority of trained architectures) and with the same fixed set of hyperparameters for each dataset. The authors of the NAS-Bench-201 use stochastic gradient descent with Nesterov momentum, batch size $N_{batch} = 256$, learning rate between 0.1 and 0 with cosine annealing and weight decay of 5×10^{-4}. Architectures are trained for 200 epochs.

Experimental scheme

The goal of this part of the study is to determine how efficiently does the CV_u^{acc} scoring metric select a good architecture among many. For this, a subset of architectures is selected at random from the benchmark dataset and their CV_u^{acc} values are evaluated by a single forward pass of a batch of training. In order to obtain statistically significant information, the random selection process is repeated $N_{runs} = 500$ times, each time choosing N_a architectures among $15,625$ available. Each architecture is initialised N_{init} times, in order to access the mean μ_u^{acc} and standard deviation σ_u^{acc} of the untrained performance. To achieve fair comparison, the data batch used for evaluation is fixed for all the architectures during the run, so that there is no uncertainty coming from the data choice. The pseudocode for this part of the study is given in Algorithm 2. For this part of study, we use a modified version of the code provided by Mellor et al. together with their paper [66].[1]

As the NAS-Bench-201 consists of automatically created architectures, some of them are constituted of meaningless combination of operations (for example, architectures consisting of skip-connection layers only). Such architectures yield constant outputs with $CV_u^{acc} = \sigma_u^{acc}/\mu_u^{acc} = 0$. As we aim to minimise CV_u^{acc}, we add a condition to filter out all the architectures with zero variance.

4.5.2 Performance of CV_u^{acc} on NAS-Bench-201

The results of the CV_u^{acc} performance with CIFAR-10, CIFAR-100 [79] and ImageNet16-120 [80] are given in Table 4.2. We present our results based on 100 initialisations ($N_{init} = 100$), for $N_{batch} = 256$, both used by Mellor et al. [66] and during the NAS-Bench-201 training. We also provide the results on the most popular NAS algorithms: regularised evolutionary algorithm (REA) [52], random search, REINFORCE [59], Bayesian optimisation and hyperband (BOHB) [81], and one-shot learning algorithms random search with parameter sharing (RSPS) [82], differentiable architecture search (DARTS) [83], gradient-based search using differentiable architecture sampler (GDAS) [84], self-evaluated template network (SETN) [85] and efficient neural architecture search (ENAS) [61]. The performance of these methods is provided by the authors

[1] The modified code can be found on GitHub at https://github.com/egracheva/TrainlessNAS_NAS201Bench

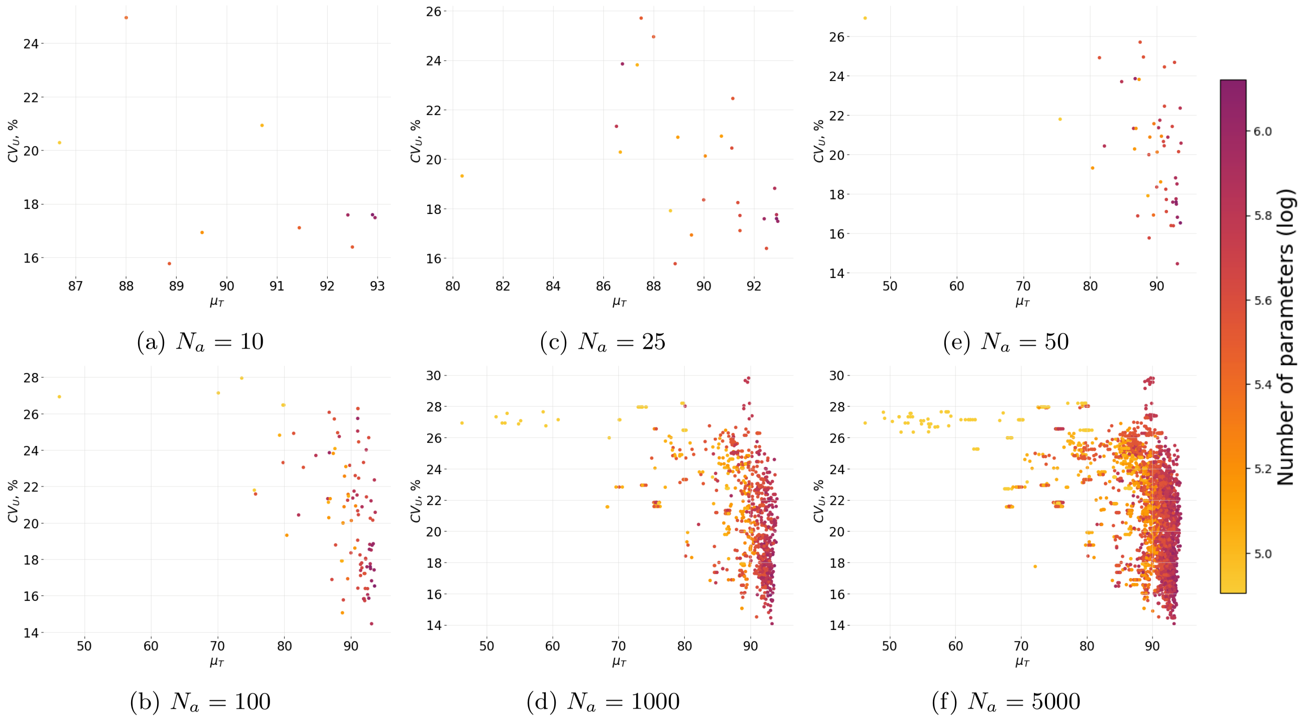
Algorithm 2 CV_u^{acc} tests on NAS-Bench-201

```
for run in range($N_{runs}$) do
    Randomly select $N_{batch}$ images from the training dataset
    Randomly select $N_a$ architectures from the whole space                ▷ arches
    for arch in arches do
        for seed in range($N_{init}$) do
            Initialise the arch with the seed
            Forward propagate selected $N_{batch}$ images
            Compute untrained accuracy $U_i$
        end for
        Compute mean $\mu_u^{acc}$, standard deviation $\sigma_u^{acc}$ for untrained accuracies over initialisations
            $\mu_u^{acc} = \frac{1}{N} \sum_{i=1}^{N_{init}} U_i, \quad \sigma_u^{acc} = \sqrt{\frac{\sum_{i=1}^{N_{init}} (U_i - \mu_u^{acc})^2}{N_{init}}}$
        Compute the score
            $CV_u^{acc} = \frac{\sigma_u^{acc}}{\mu_u^{acc}}$                (4.1)
    end for
    Select the architecture with the minimum score value ($CV_u^{acc} > 0$)
    Retrieve trained accuracy $T$ for the selected architecture from the database
end for
Average trained accuracies of selected architectures over $N_{runs}$
    $\mu_t = \sum_{j=1}^{N_{runs}} T_j$                (4.2)
```

of NAS-Bench-201 benchmark [67].

The results show that the performance of the CV_u^{acc} scoring metric is clearly above random for all three datasets.

The effects of number of iterations and number of selected architectures are shown in Figures 4.6 and 4.7, respectively, on an example of CIFAR-10 [79]. The number of picked architectures considerably increases the overall performance, since there is more chance to involve a good architecture. The number of iterations improves the precision of



(a) $N_a = 10$

(c) $N_a = 25$

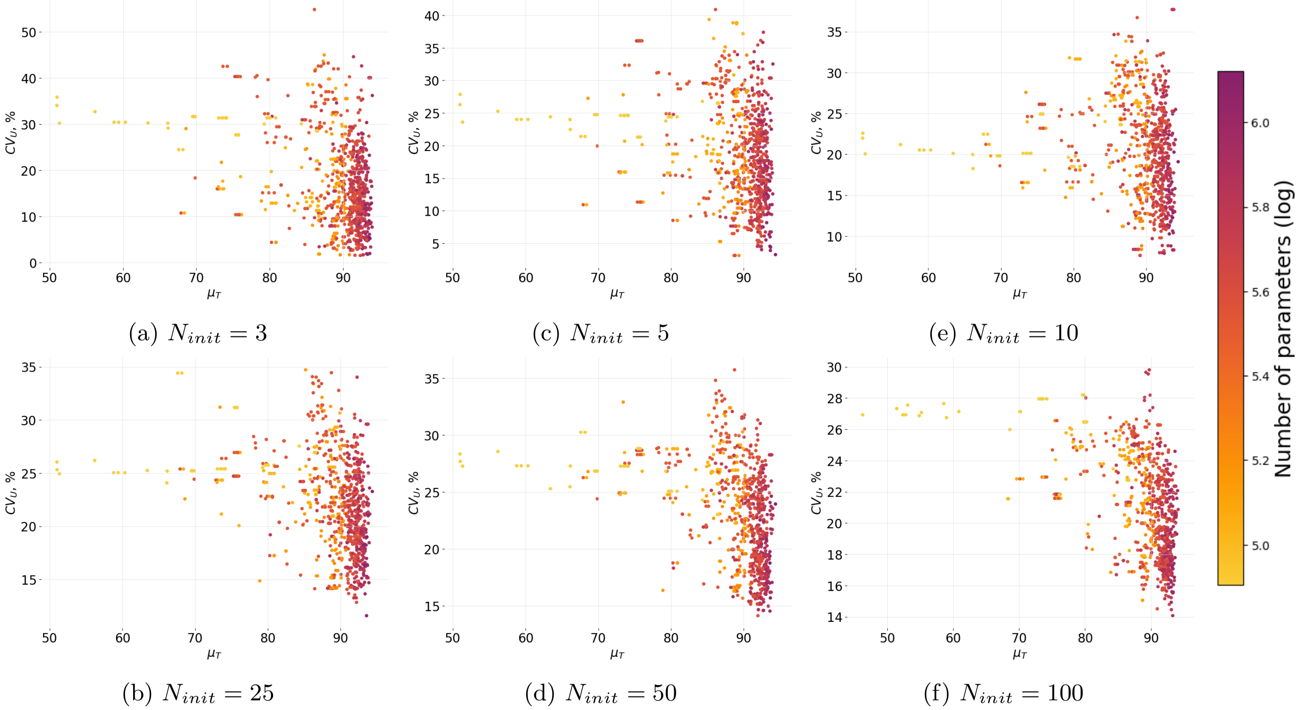
(e) $N_a = 50$

(b) $N_a = 100$

(d) $N_a = 1000$

(f) $N_a = 5000$

FIGURE 4.6: Comparison of the coefficient of variation CV_u^{acc} performance against mean trained accuracy μ_T for CIFAR-10 dataset for different number of selected architectures $N_a \in [10, 25, 50, 100, 1000, 5000]$. Statistics are computed over $N_{init} = 100$ initialisations. One point stands for one architecture. The colours represent the logarithm of the total number of trainable parameters.



(a) $N_{init} = 3$

(c) $N_{init} = 5$

(e) $N_{init} = 10$

(b) $N_{init} = 25$

(d) $N_{init} = 50$

(f) $N_{init} = 100$

FIGURE 4.7: Comparison of the coefficient of variation CV_u^{acc} performance against mean trained accuracy μ_T for CIFAR-10 dataset. Statistics are computed over varying number of initialisations $N_{init} \in [3, 5, 10, 25, 50, 100]$. Number of architectures $N_a = 1000$. One point stands for one architecture. Colours represent the logarithm of the total number of trainable parameters.

Method	Time(s)	CIFAR-10		CIFAR-100		ImageNet16-120	
		validation	test	validation	test	validation	test
State-of-the-art							
REA	12000	91.19 ± 0.31	93.92 ± 0.3	71.81 ± 1.12	71.84 ± 0.99	45.15 ± 0.89	45.54 ± 1.03
Random Search	12000	90.93 ± 0.36	93.92 ± 0.31	70.93 ± 1.09	71.04 ± 1.07	44.45 ± 1.1	44.57 ± 1.25
REINFORCE	12000	91.09 ± 0.37	93.92 ± 0.32	71.61 ± 1.12	71.71 ± 1.09	45.05 ± 1.02	45.24 ± 1.18
BOHB	12000	90.82 ± 0.53	93.92 ± 0.33	70.74 ± 1.29	70.85 ± 1.28	44.26 ± 1.36	44.42 ± 1.49
One-shot learning							
RSPS	7587	84.16 ± 1.69	84.07 ± 1.69	59.00 ± 4.60	58.33 ± 4.34	31.56 ± 3.28	31.14 ± 3.88
DARTS-V1	10890	39.77 ± 0.00	54.30 ± 0.00	15.03 ± 0.00	15.61 ± 0.00	16.43 ± 0.00	16.32 ± 0.00
DARTS-V2	29902	39.77 ± 0.00	54.30 ± 0.00	15.03 ± 0.00	15.61 ± 0.00	16.43 ± 0.00	16.32 ± 0.00
GDAS	28926	90.00 ± 0.21	93.51 ± 0.13	71.14 ± 0.27	70.61 ± 0.26	41.70 ± 1.26	41.84 ± 0.90
SETN	31010	82.25 ± 5.17	86.19 ± 4.63	56.86 ± 7.59	56.87 ± 7.77	32.54 ± 3.63	31.90 ± 4.07
ENAS	13315	39.77 ± 0.00	54.30 ± 0.00	15.03 ± 0.00	15.61 ± 0.00	16.43 ± 0.00	16.32 ± 0.00
Baselines							
Optimal ($N_a = 100$)	N/A	91.05 ± 0.28	93.84 ± 0.23	71.45 ± 0.79	71.56 ± 0.78	45.37 ± 0.61	45.67 ± 0.64
Optimal ($N_a = 1000$)	N/A	90.35 ± 0.15	94.20 ± 0.13	72.54 ± 0.52	72.83 ± 0.39	45.88 ± 0.55	46.60 ± 0.33
Random	N/A	83.20 ± 13.28	86.61 ± 13.46	60.70 ± 12.55	60.83 ± 12.58	33.34 ± 9.39	33.13 ± 9.66
Trainless							
NASWOT ($N_a = 100$)	30.0	89.55 ± 0.89	92.81 ± 0.99	69.35 ± 1.70	69.48 ± 1.70	42.81 ± 3.05	43.10 ± 3.16
NASWOT ($N_a = 1000$)	306.2	89.69 ± 0.73	92.96 ± 0.81	69.86 ± 1.21	69.98 ± 1.22	43.95 ± 2.05	44.44 ± 2.10
CV_u^{acc} ($N_a = 100$)	171.4	84.89 ± 6.39	91.90 ± 2.27	63.99 ± 5.61	64.08 ± 5.63	38.68 ± 6.34	38.76 ± 6.62

TABLE 4.2: Comparison of the relative standard deviation on the untrained accuracy, CV_u^{acc} (%), performance against existing NAS algorithms on CIFAR-10, CIFAR-100 and ImageNet16-120 datasets. On the top, the best performing methods that require training are listed (REA, random search, REINFORCE, BOHB). Then, the NASWOT and our results are reported for $N_a \in \{100, 1000\}$ with $N_{batch} = 256$. Elapsed times are reported as median times among three datasets. Random and optimal values for $N_a \in \{100, 1000\}$ are given as baseline.

the method. Similar plots for CIFAR-100 [79] and ImageNet16-120 [80] can be found in Appendix (Figures 6.1, 6.2, 6.3, 6.4). Table 4.2 shows results of our metric performance with various N_{batch}, N_{init} and N_a combinations.

We compare our results against the performance of the NASWOT presented by Mellor et al., since this method also aims to discover purely architectural property responsible for good network trainability. We do not draw a comparison with quasi-trainless NAS methods, as they rely on a supplementary model responsible for the architecture choice and therefore lack interpretability. In Table 4.2, similar overall performances are observed. According to the obtained results, both our metric and NASWOT are also similar in the sense that they filter out bad architectures, rather than choose the best one.

Our approach focuses on how much variation in weights affects the outputs. CV_u^{acc} quantifies the stability of the network against initialisations for the same fixed data minibatch. Intuitively, if a network is stable against random weights, it will also be less affected by weights fluctuations during the training. It might suggest that the function representing a stable network is relatively smooth, which allows for more efficient training and lower overfitting risks.

The downside of our algorithm is that it involves two extra hyperparameters. The first one is the BS: there are significant deviations on the prediction power (with different optimal N_{batch} for each dataset, see Table 6.3). The second is the number of initialisations. Besides, the fact that our method requires multiple initialisations leads to a significantly slower performance compared to NASWOT (running time grows linearly with the number of initialisations). Yet, comparing to the methods that require training, the absolute performance speed remains high (tens to hundreds of seconds).

We can also see that the prediction accuracy improves with the number of sampled architectures (for any BS). This is a natural consequence of the fact that the chance of having a well performing architecture is higher when there are many available architectures (which is confirmed by random selection tests, see Table 4.2).

Overall, it is clear that the CV_u^{acc} metric alone is not sufficient for successful NAS. While there is a possibility that the architectures selected by our metric could have achieved better accuracies if trained with optimal hyperparameters, it is obvious that the method needs improvement. Nevertheless, achieved results lead us to the conclusion that the stability of a network against initialisations is an indicator of its trainability.

4.5.3 Ways to improve the CV_u^{acc} metric

While providing important insights on the ability of NNs to fit the data, the CV_u^{acc} comes with the following shortcomings.

First, accuracy-based scoring metric can only be applied to classification problems, and it is not clear how to extend the above results to regression tasks. To make the method applicable to every problem, we need to look at some other performance measure. Besides it is clear that the precision of the metric is far from being optimal. This can be partly explained by the fact that random weights initialisation brings in some noise. We will address these issues in our future work.

4.6 Conclusions on Chapter 4

In this part of the thesis we explore relationship between the prediction performance of an architecture and its performance prior to training. The principal objective is to better understand how the NN's geometry affects its prediction power. For this, we explore untrained accuracy over multiple random weights initialisations. We observe that the architectures with low coefficient of variation of untrained accuracy $CV_u^{acc} = \sigma_u^{acc}/\mu_u^{acc}$ show overall better performance.

We use these observations to develop an entirely trainless NAS technique. We introduce a fully trainless NAS procedure. On NAS-Bench-201 [67], our metric achieves the accuracies of 91.90 ± 2.27, 64.08 ± 5.63 and 38.76 ± 6.62 for CIFAR-10, CIFAR-100 [79] and a downscaled version of ImageNet [80], respectively (when choosing among 100 architectures, with 5 different weights on a minibatch of 256 data points).

The success of CV_u^{acc} allows us to make the conclusion that stability of a network against initialisations is an indicator of its trainability.

We acknowledge here that a part of the study in this chapter is published as [P2] in the list of publications.

5. Prediction of Polymer Thermal Properties

5.1 Modified SMILES-X

In the present Chapter we put together the SMILES-X machine learning tool described in Chapter 3 and NAS technique developed in Chapter 4 to build predictive models for coefficient of linear thermal expansion, β. We also discuss some auxiliary changes in the SMILES-X software that were made to achieve the most reliable results.

5.1.1 Implementation of trainless NAS

Here we discuss the implementation of the trainless metric into the SMILES-X. While the overall architecture of the SMILES-X is fixed, it remains to define the optimal number of units in the LSTM, dense and embedding layers (the skeleton of SMILES-X was given in Figure 3.2). As we described before, in the original version of the software, neural geometry and hyperparameters required for training (BS and LT) were determined via BO. Now we split the process into two independent steps, since we succeeded to make the geometry search independent from BS and LT. Therefore, the implementation of trainless NAS brought the following key improvements.

First, since the search of the optimal architecture geometry can now be performed independently from the BS and LT, a single complex 5-dimensional search space became divided into two considerable smaller individual search spaces of 3 and 2 dimensions, for architecture search and training hyperparameters search, respectively. In the beginning, geometry hyperparameters are fixed by minimising the CV_u^{acc} score, then BO is used to define optimal training hyperparameters. This allows BO to operate in a space of a smaller dimensionality, and therefore to increase relative coverage within this space in the same amount of time.

Second, introduction of trainless NAS has enabled the SMILES-X to find a well fitting and stable predictive model for each of the properties in a fully automatic manner. Before, BO alone could not find a good combination of hyperparameters – recommended architectures could not fit even the training data. Some randomly selected architectures could to some extent fit the training data, but were extremely unstable depending on the data split and initialisation weights. Architectures found via trainless NAS fit the data regardless the seed and data split (smooth learning curves, 100% convergence).

5.2 Modification of Bayesian optimisation search

For the BO step we have slightly modified the procedure comparing to the original SMILES-X package. The original BO optimises parameters based on the best validation RMSE over the first 50 epochs. However, when a dataset becomes too small (of the order of 100 data points or less), the validation set becomes extremely small as well. This circumstance naturally leads to very noisy learning curves. Also, for some combinations of LT and BS, learning curves diverge after 50 epochs have passed. These two factors cause the originally implemented BO process to make faulty conclusions on the shape of the underlying function $f_{\mathrm{RMSE}}(LR, BS)$, and consequently to result in a poor choice of

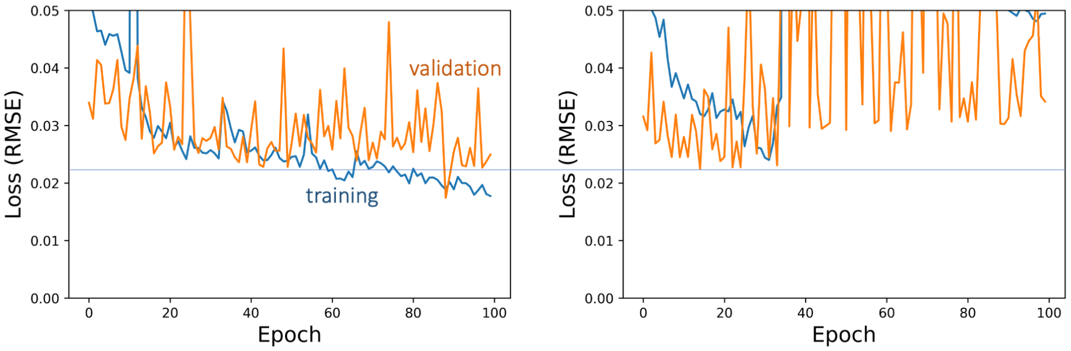


Figure 5.1: Learning curves for two different combinations of hyperparameters during Bayesian optimisation within the SMILES-X neural architecture. Orange and blue lines represent the evolution of validation and training root-mean-squared error, respectively. High amount of noise is attributed to the small size of data. Horizontal line shows minimum value achieved by architectures within 50 epochs.

LT and/or BS. These effects are illustrated in Figure 5.1). Looking at the overall trend of the learning curves we can conclude that the left model is preferable and should be chosen, and the right one diverges and should be rejected. However, if we take decision based on the lowest validation error within 80 epochs of training, the right one would be selected.

To counteract these two issues, as discussed above, we extended the training during BO to 100 epochs, and the function values are based on the mean RMSE over 10 runs, with random data splitting between training and validation at every run. The overall procedure takes about 20 times longer than in the original SMILES-X version. Yet this duration is not computationally prohibitive, since the data is small.

Even though the overall hyperparameters optimisation procedure did not become faster, it became considerably more efficient. Most of the time is taken to optimise training hyperparameters search via BO, since this part inevitably requires model training. Therefore, overall introduction of trainless NAS has mainly reduced and simplified the search space for the BO. This in turn allows BO to cover more search space in the same amount of time and to output results of higher confidence.

	NAS			BO	
Property	n_{lstm}	n_{dense}	n_{embed}	BS	LT
β	128	128	512	8	$10^{-3.4}$

Table 5.1: Optimal number of units in the LSTM, dense and embedding layers (n_{lstm}, n_{dense}, n_{embed}) for the SMILES-X neural architecture found via trainless NAS method, together with optimal BS and LT proposed by BO for coefficient of linear thermal expansion, β.

5.2.1 Improved NAS performance and results

During the geometry search, we evaluate architectures on a fixed set of 16 randomly selected SMILES. As extremely small and extremely large weights lead to divergence of the signal within the network (NaN outputs), we set the shared constant weight values within the range of [1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1, 10, 100].

Hyperparameters search is performed only for the first fold. The same hyperparameters are then used for the remaining folds. The NAS and BO procedures took 2 and 15.5 hours, respectively, and the set of chosen hyperparameters is given in Table 5.1.

5.3 Random forest

While our research is focused on the NLP approach in molecular properties prediction, we build a RF model with fingerprint inputs as a baseline proxy. As it was mentioned in the introductory part (Section 3.2), the choice of fingerprint for the RF is task dependant and requires extensive field expertise. As the aim of RF model is to provide a baseline for the SMILES-X models, we do not put much focus on the fingerprint selection. We take all the available fingerprints within a well known RDKit python library [86] and try their performances on β dataset. The best one is used for the final model training. The full list of fingerprints is given in Table 5.2 and the best fingerprint is indicated with circle.

Fingerprint	β
Morgan	
RDkit	○
Atom pairs	
Topological torsion	
Molecular access system (MACCS)	
Extended reduced graph approach (Erg)	

TABLE 5.2: The list of fingerprints used for RF training. The best performing fingerprint on the β dataset is denoted with a circle.

RF also requires to set the hyperparameters for training. Here we use the default values provided by RandomForestRegressor function within the sklearn library [87]:

```
n_estimators=100                  min_weight_fraction_leaf=0.0
max_depth=None                    min_impurity_decrease=0.0
max_samples=None                  criterion='squared_error'
bootstrap=True                    min_samples_split=2
oob_score=False                   min_samples_leaf=1
ccp_alpha=0.0                     max_features='auto'
warm_start=False                  max_leaf_nodes=None
n_jobs=None                       random_state=None
verbose=0
```

Similar to SMILES-X, RF models are trained 5 times with different random seeds and train/validation splits.

5.4 Data

The data used for the model training is retrieved from the polymer database PoLyInfo [88] belonging to the National Institute for Materials Science (Japan) – the largest available polymer database to date. To keep the data consistent, we perform meticulous data selection. Nevertheless, it is important to keep in mind that experimental values are associated with large systematic errors, since measured thermal properties depend on the measuring technique and experimental setup.

For the CTE, linear coefficient β is easier to measure than volumetric α, which is why it became the most often reported expansion measure in the scientific literature. It is also the most represented within the PoLyInfo database [88]. Even though in case of anisotropic polymers single dimension measurement is not representative of the overall material behaviour, linear expansion reported in an arbitrary direction is generally considered to represent the behaviour of the sample as a bulk.

Out of the initial 1580 entries for β available within the PoLyInfo [88], we keep only amorphous homopolymers in glassy state. Unlike homopolymers, co-polymers consist of multiple types of repeating units. As there are countless ways to combine these repeating units within the chain, copolymers come with a lot of uncertainty. Amorphous type is chosen because material expansion changes with the degree of crystallinity, and it is often omitted in the scientific literature. Amorphous polymers, on the other hand, have the degree of crystallinity close to 0, reducing the amount of uncertainty comparing to crystalline polymers. Glassy state is chosen for consistency reasons: since polymer thermal properties differ between glassy and rubbery states, it is important to use only one of the regions. Otherwise, there is no difference between the states in terms of ease of prediction, so the glassy state is chosen arbitrarily. Next, we remove all the samples with fillers, as we want the material to be entirely defined by the repeating unit only. Finally, we remove all the samples in form of ultra-thin films: atoms within the materials of thickness <1 µm have reduced number of degrees of freedom and show anisotropic thermal expansions.

While the data selection is mainly performed automatically, many of the entries with missing information were manually verified. We had to confirm, for example, whether a given measurement has been performed below or above the glass transition temperature, as this information is often unreported within the database. In case of multiple measurements reported for the same monomer, the median value is used for training. For β, the overall data selection procedure results in a total of 106 unique repeating units

Property	Model	R^2-score	RMSE	MAE
β ($10^{-5}\,\mathrm{K}^{-1}$)	SMILES-X	0.68 ± 0.06	2.44 ± 0.17	1.71 ± 0.10
	Random forest	0.65 ± 0.02	2.54 ± 0.07	1.57 ± 0.06

Table 5.3: Out-of-sample R^2-score, RMSE and MAE of the trained SMILES-X and RF ensemble models on the coefficient of linear thermal expansion, β ($10^{-5}\,\mathrm{K}^{-1}$). The reported mean is calculated over 5 models trained with different random seeds. The error corresponds to a single standard deviation.

Property	Time	
	SMILES-X	RF
β	26:22.39	00:00.08

Table 5.4: Average training times per run for SMILES-X and RF models for for training after data selection process for the coefficient of linear thermal expansion, β. Times are reported in format *mm:ss.cs*.

associated with median values.

5.5 Prediction of thermal properties for amorphous homopolymers

The out-of-sample predictions of the trained SMILES-X and RF ensemble models for the coefficient of linear expansion, β, is given in Figure 5.2. Quantitative prediction performances are summarised in Table 5.3. The models are based on 20-fold cross validation. Average training times per run are given in Table 5.4, with RF taking less than a second per run, and SMILES-X taking about 26 minutes on a single GPU.

The SMILES-X model shows a very satisfactory overall agreement with the experimental data. It is also encouraging to see that many data points have similar predictions for the two distinct ML approaches.

For β properties predicted by SMILES-X, the reported prediction means and errors are computed by averaging both over $\mathrm{N_{augm}}$ augmented SMILES and $\mathrm{N_{mod}} = 5$ random initialisations as follows:

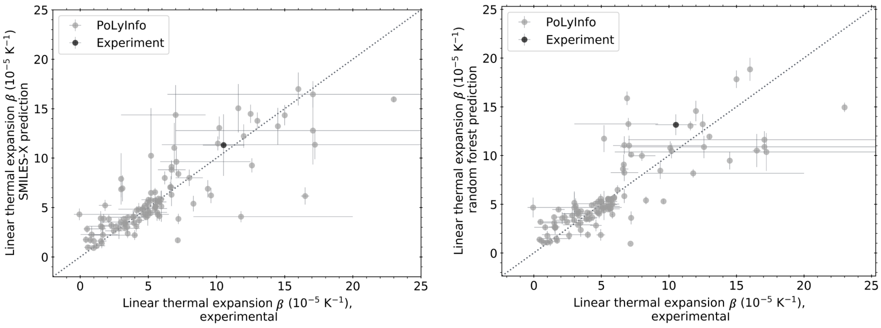


FIGURE 5.2: The out-of-sample predictions (y-axis) for the coefficient of linear thermal expansion, β ($10^{-5}\,\mathrm{K}^{-1}$), given by (a) SMILES-X [P1] and (b) RF ensemble models against the experimental data (x-axis). Grey points represent the median of the values provided within the PoLyInfo dataset. The black point represents the mean of the experimentally measured poly(vinyl methyl ketone) sample. Error bars on the x-axis span between experimental minimum and maximum values, and error bars on the y-axis represent one standard deviation from the distribution of the predicted values. Prediction values indicate the mean over 5 trained models.

$$\begin{aligned}\mu(\beta_{pred}) &= \frac{\sum\limits_{i=1}^{N_{\text{mod}}}\sum\limits_{j=1}^{N_{\text{augm}}}\beta_{ij}}{N_{\text{mod}}\cdot N_{\text{augm}}},\\ \sigma(\beta_{pred}) &= \sqrt{\frac{\sum\limits_{i=1}^{N_{\text{mod}}}\sum\limits_{j=1}^{N_{\text{augm}}}(\beta_{ij}-\mu(\beta_{pred}))^2}{N_{\text{mod}}\cdot N_{\text{augm}}}},\end{aligned} \tag{5.1}$$

where β_{ij} corresponds to a single model prediction on a single SMILES.

As RF does not principally implement data augmentation, prediction means and errors are computed based on $\mathrm{N}_{\mathrm{mod}} = 5$ random initialisations only:

$$\begin{aligned}\mu(\beta_{pred}) &= \frac{\sum\limits_{i=1}^{N_{\text{mod}}}\beta_{i}}{N_{\text{mod}}},\\ \sigma(\beta_{pred}) &= \sqrt{\frac{\sum\limits_{i=1}^{N_{\text{mod}}}(\beta_{i}-\mu(\beta_{pred}))^2}{N_{\text{mod}}}},\end{aligned} \tag{5.2}$$

where β_i corresponds to a single model prediction.

The overall out-of-sample prediction shows a RMSE of $2.65 \pm 0.09 \times 10^{-5}\,\mathrm{K}^{-1}$ ($2.61 \pm 0.11 \times 10^{-5}\,\mathrm{K}^{-1}$), a mean absolute error of $1.71 \pm 0.06 \times 10^{-5}\,\mathrm{K}^{-1}$ ($1.68 \pm 0.07 \times 10^{-5}$ K^{-1}) and a coefficient of determination of 0.62 ± 0.03 (0.63 ± 0.03) for SMILES-X (RF). Tables 6.2 and 6.1 in Appendix provide details on fold-wise performances for the SMILES-X and RF, respectively. Both models show fair amount of the variability in the predictions depending on the fold. This can be explained by the modest size of the dataset: random splitting into training, validation and test sets does not guarantee that the three sets will have the same proportions of polymer types.

It is known that the CTE of a polymer varies with molecular weight $\mathrm{M_W}$ [89], therefore the accuracy of our method is limited to the amount of variation of the CTE values between extreme $\mathrm{M_W}$ values. Assuming that most of the measurements accumulated within the PoLyInfo database are performed in the range of average molecular weights, it is not surprising that the models can make relatively good predictions even without the $\mathrm{M_W}$ information.

5.6 Further model validation for the coefficient of linear thermal expansion

For the coefficient of linear thermal expansion, β, we perform a deeper validation of the model. We do it in three planes: via experimental validation, analysing attention maps and comparing to another existing semi-empirical method.

5.6.1 Experimental validation

While the presented prediction results already show out-of-sample performance of the predictive ensemble models, we further test them with an experimental sample prepared in our laboratory. For this, we have selected 9 commercially available polymers having no β entries within the PoLyInfo [88]. After processing the polymer powder into a film shape, 8 samples out of 9 did not meet the criteria for the measurement: 5 films had high air bubble content, and 3 of the remaining 4 films were too brittle to conduct the measurement. Therefore, we test the model with a single sample – poly(vinyl methyl ketone). The raw poly(vinyl methyl ketone) powder is purchased at Sigma-Aldrich (average $\mathrm{M_w} \sim 500,000$, $\mathrm{T_g} \sim 28\,^{\circ}\mathrm{C}$).

The film is prepared using a hot press following the below procedure. About 1 g of the poly(vinyl methyl ketone) powder is deposed on a metal plate sprayed with silicon, with two metal spacers of 0.5 mm used for the thickness control. The plate is then placed onto the bottom surface of a preheated hot press (80 °C). Once the powder shows signs of transition (starts melting), the second metal plate is placed on top, and a pressure of 8 MPa is applied for 1 min. The plates are cooled down in water ice, and the sample is then extracted from between the plates. The shape of the resulting sample is nearly

Sample #	β ($10^{-5}\,\mathrm{K}^{-1}$)
1	8.16
2	11.43
3	12.16
4	10.32
Overall	10.52 ± 1.51

Table 5.5: The experimental results for the coefficient of linear thermal expansion, β ($10^{-5}\,\mathrm{K}^{-1}$) measured on a poly(vinyl methyl ketone) sample. Overall, the mean value with one standard deviation are reported.

circular, with a diameter of about 60 mm and a thickness of 5 mm.

The CTE measurement is performed in extension mode using a thermomechanical analyzer Rigaku TMA8311. A sample is applied at a constant load of 49 mN and extension was recorded while increasing sample temperature at a constant rate of 5 C/min under nitrogen flow of 50 ml/min. Four rectangular subsamples (20 mm × 4 mm) are cut off from the original sample and measured independently by thermomechanical analysis. The detailed information on experimental setup and temperature graphs can be found in Appendix.

It is worth noting that while all the four samples have been cut out of the same sample and measured by the same equipment under the same environmental conditions, there exist large deviation between the resulting values. Relative standard deviation is of $\sim 14\%$. This can be explained by uneven air bubble distribution or uneven thickness of the lab prepared film. Nonetheless, it is a good reminder of the precision of the data that can be found in materials science databases.

Table 5.6 shows predictions from SMILES-X and RF ensemble models, each composed of 100 trained models (which corresponds to all of the trained models, with 5 runs and 20 folds). When comparing against the experimental results, we can see very good agreement (p-value $\gg 0.05$ for both methods). Furthermore, poly(vinyl methyl ketone) has a relatively high expansivity, while most of the data used for models training comes

	β ($10^{-5}\,\mathrm{K}^{-1}$)		p-value
	Experiment	Prediction	
SMILES-X	10.52 ± 1.52	11.16 ± 1.52	0.47
Random forest		10.43 ± 1.78	0.92

TABLE 5.6: Comparison of the coefficient of linear thermal expansion, β ($10^{-5}\,\mathrm{K}^{-1}$) values predicted by SMILES-X and RF ensemble models against experimental measurement for a poly(vinyl methyl ketone). For the predictions, mean value and one standard deviation are reported over 100 trained models, corresponding to 5 runs and 20 folds. For the experiment, statistics are reported over 4 measurements.

from polymers with relatively low CTE. This is an indicator that the models are well balanced and can be used to replace the experiment even in the regions with fewer data.

5.6.2 Interpretation

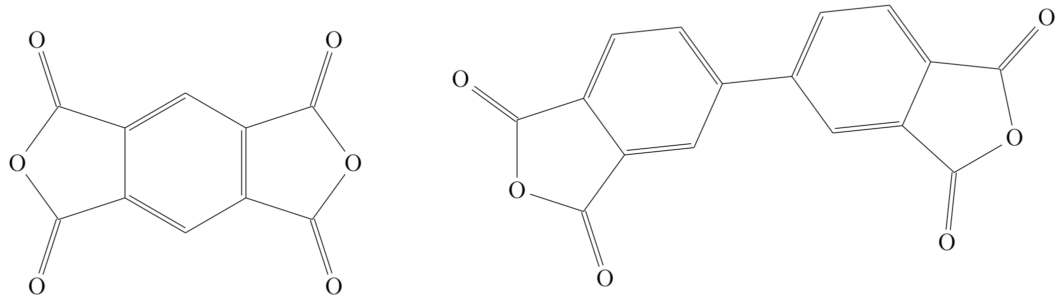
The attention mechanism implemented in the SMILES-X permits to have a visual comprehension of which atom or bond a trained model pays attention to when computing β. Note that single bonds and hydrogen atoms are not represented by the SMILES used in this thesis. Molecular fingerprint based RF model allows a similar kind of visualisation. However, due to the nature of fingerprint computation, it is only possible to evaluate the importance of individual atoms towards the prediction, and not of branches or bonds. We present below, in Figures 5.4-5.7, attention maps for some of the studied homopolymers. These maps show out-of-sample attention, *i.e.* they correspond to the fold where a given polymer appears in the test set and is not seen by the model during training. Similar to predictions, they are a result of averaging over $\mathrm{N}_{\mathrm{mod}} = 5$ random initialisations for RF, as well as over N_{augm} augmented SMILES for the SMILES-X (Equations 5.2 and 5.1). Note, that for the SMILES-X attention maps vary between different SMILES representations for the same molecule (see Figure S1 in the Supplementary materials for an example).

The main known factors that affect the CTE of a homopolymer are cohesive forces between chains, topological and geometrical arrangement of atoms, chain stiffness and bond flexibility [17]. For example, alkyl chains are flexible and show high values of the CTE. As demonstrated in Figure 5.4a, the SMILES-X model has successfully learnt this feature from the data, while RF shows that the most important atoms are sitting on the very end of the chain.

Another example is the inclusion of double or triple bonds in the homopolymer chains. These are known to be rigid structures showing little or no rotation, thus obtruding the movement of polymer chains within the bulk material and therefore lowering the CTE. However, there are only two samples containing triple bonds within our dataset, and, unsurprisingly, the model does not pay much attention to this component. Figures 5.4b, 5.4c demonstrate performance of both ensemble models on such polymers.

Polyimides, due to their inherently low thermal expansion, are of great interest for the industry and attract a lot of attention in research and development, which makes this class to be the most represented within our CTE dataset. Particularly, pyromellitic dianhydride (PMDA) and biphenyltetracarboxylic dianhydride (BPDA) groups consist of multiple rings, which makes their structures very rigid (see Figure 5.3). Accordingly, both presented machine learning models successfully associate polyimides with low CTE values, and the SMILES-X model pays attention to PMDA and BPDA structures. as seen in Figures 5.5a, 5.5b. While the SMILES-X rather pays attention to the double bonds and nitrogens characteristic to polyimides, the RF is focused on other structures instead.

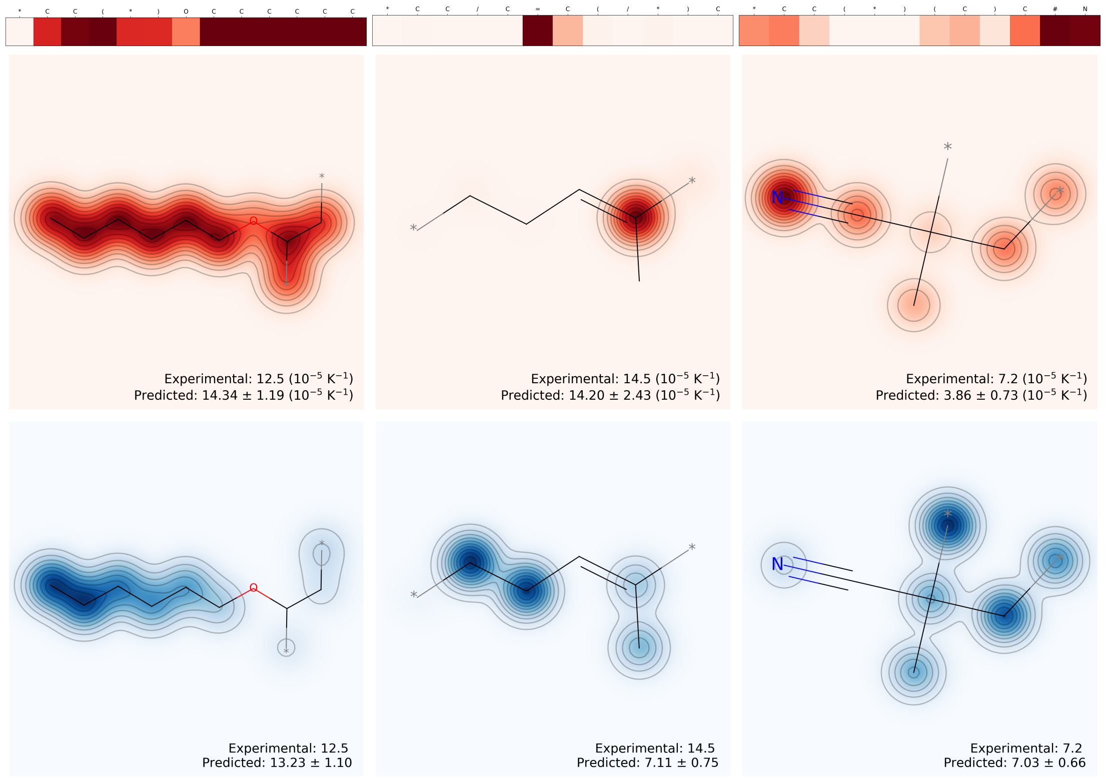
Note also that the SMILES-X attention is rather paid to branches and bonds than to the atoms themselves, as shown in Figure 5.5c. This reflects the intuition that the shape of a repeating unit is one of the most important features influencing the CTE. Nevertheless, molecular fingerprints contained hashed information about every atom's environment including the bond information, so while RF model is incapable to point out bonds themselves, it may point out the atom in the vicinity of an important bond.



PMDA

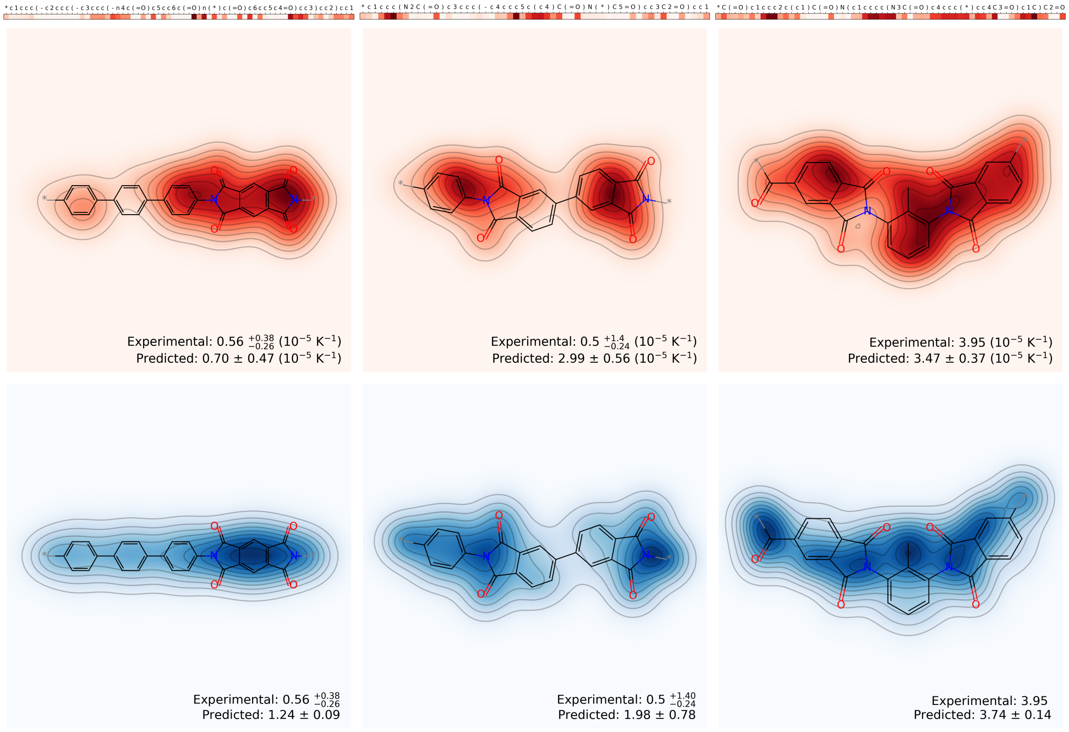
BPDA

Figure 5.3: Graph representations of PMDA (left) and BPDA (right) chemical structures


* C C (*) O C C C C C C
* C C / C = C (/ *) C
* C C (*) (C) C # N
O
N
Experimental: 12.5 (10^{-5} K^{-1})
Predicted: 14.34 ± 1.19 (10^{-5} K^{-1})
Experimental: 14.5 (10^{-5} K^{-1})
Predicted: 14.20 ± 2.43 (10^{-5} K^{-1})
Experimental: 7.2 (10^{-5} K^{-1})
Predicted: 3.86 ± 0.73 (10^{-5} K^{-1})
Experimental: 12.5
Predicted: 13.23 ± 1.10
Experimental: 14.5
Predicted: 7.11 ± 0.75
Experimental: 7.2
Predicted: 7.03 ± 0.66

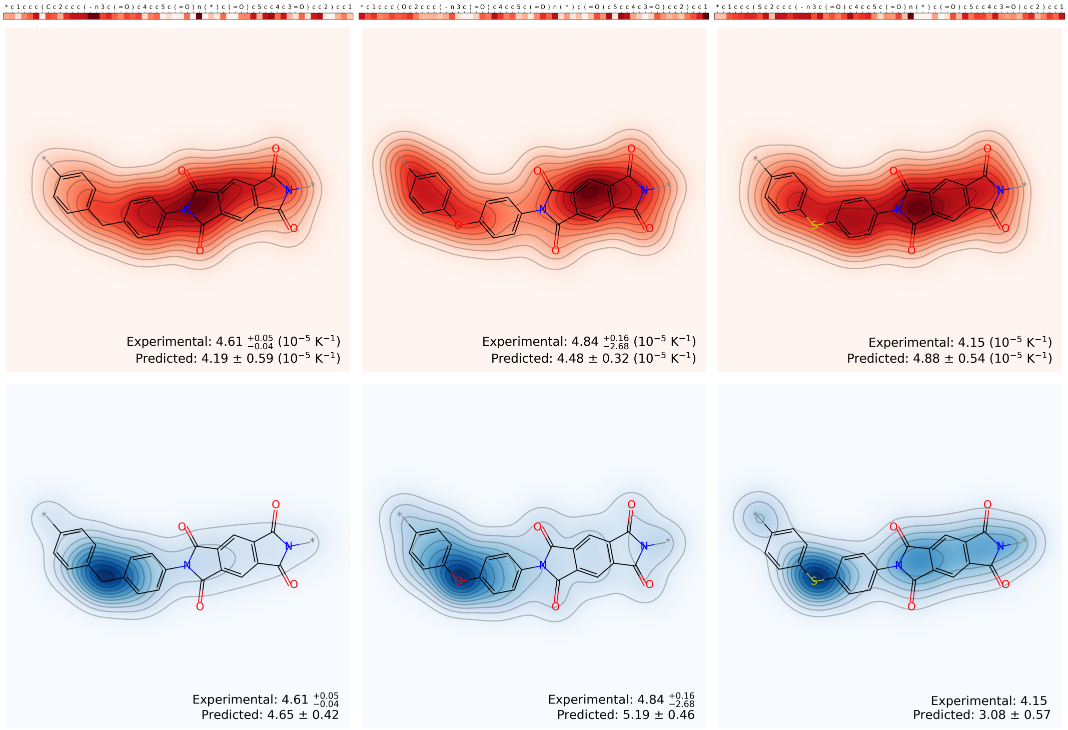
(a) (b) (c)

FIGURE 5.4: Attention maps built upon the SMILES-X (red) and RF (blue) predictive ensemble models for coefficient of linear thermal expansion, β ($10^{-5}\,\mathrm{K}^{-1}$), for homopolymers containing (a) alkyl chains (poly(decyl vinyl ether)); (b) double bonds (poly(isobutyl methacrylate)); (c) triple bonds (poly(methacrylonitrile)). For SMILES-X top and bottom rows show 1D and 2D attention maps, respectively. For RF fingerprint similarity maps are shown. The darker the shade of the colour is, the stronger is the attention paid by the respective ensemble model. At the bottom of each figure, the experimental results are given, and the predictions are given as mean with one standard deviation.



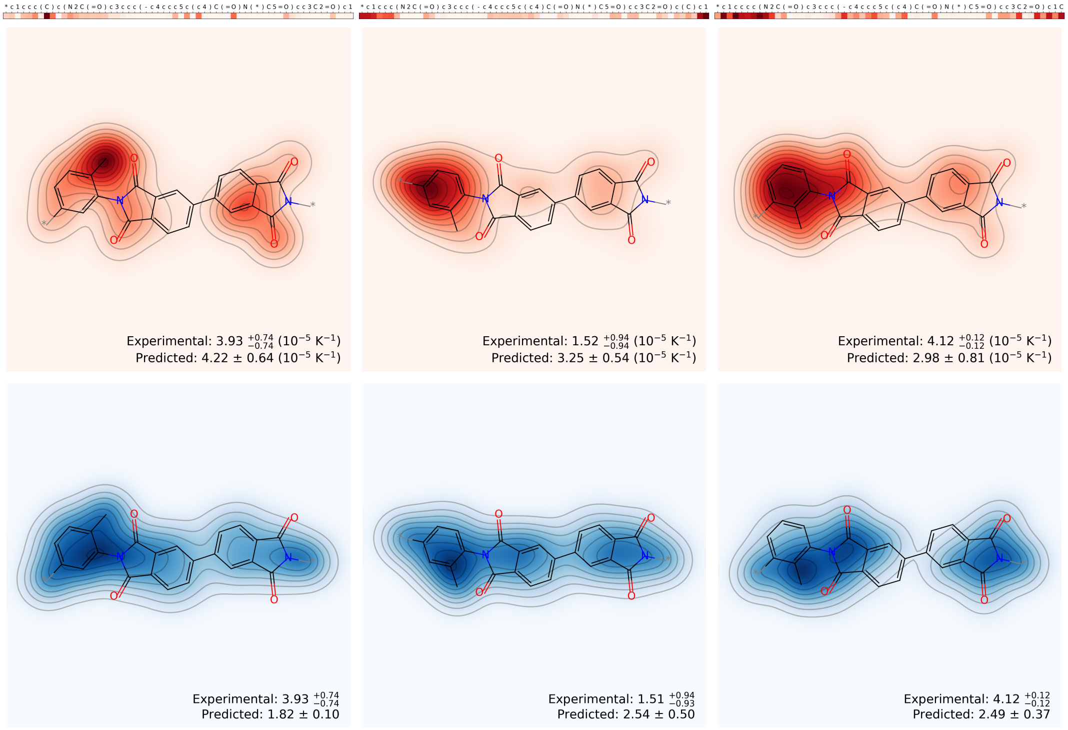
(a) (b) (c)

FIGURE 5.5: Attention maps built upon the SMILES-X (red) and RF (blue) predictive ensemble models for coefficient of linear thermal expansion, β ($10^{-5}\,\mathrm{K}^{-1}$), for homopolymers containing (a) PMDA group (poly[(1,1':4',1"-terphenyl-4,4"-diamine)-alt-(pyromellitic anhydride)]); (b) BPDA group (poly[(4,4'-oxydianiline)-alt-(biphenyl-3,3':4,4'-tetracarboxylic dianhydride)]); (c) complex shape (poly[1,2-bis(4-aminophenyl)benzene)]-alt-(biphenyl-3,3',4,4'-tetracarboxylic dianhydride)). For SMILES-X top and bottom rows show 1D and 2D attention maps, respectively. For RF fingerprint similarity maps are shown. The darker the shade of the colour is, the stronger is the attention paid by the respective ensemble model. At the bottom of each figure, the experimental results are given as median with a range of available values, and the predictions are given as mean with one standard deviation.


*c1ccc(Cc2ccc(-n3c(=O)c4cc5c(=O)n(*)c(=O)c5cc4c3=O)cc2)cc1
*c1ccc(Oc2ccc(-n3c(=O)c4cc5c(=O)n(*)c(=O)c5cc4c3=O)cc2)cc1
*c1ccc(Sc2ccc(-n3c(=O)c4cc5c(=O)n(*)c(=O)c5cc4c3=O)cc2)cc1
Experimental: 4.61 $^{+0.05}_{-0.04}$ (10^{-5} K^{-1})
Predicted: 4.19 ± 0.59 (10^{-5} K^{-1})
Experimental: 4.84 $^{+0.16}_{-2.68}$ (10^{-5} K^{-1})
Predicted: 4.48 ± 0.32 (10^{-5} K^{-1})
Experimental: 4.15 (10^{-5} K^{-1})
Predicted: 4.88 ± 0.54 (10^{-5} K^{-1})
Experimental: 4.61 $^{+0.05}_{-0.04}$
Predicted: 4.65 ± 0.42
Experimental: 4.84 $^{+0.16}_{-2.68}$
Predicted: 5.19 ± 0.46
Experimental: 4.15
Predicted: 3.08 ± 0.57

(a) (b) (c)

FIGURE 5.6: Attention maps built upon the SMILES-X (red) and RF (blue) predictive ensemble models for the coefficient of linear thermal expansion, β ($10^{-5}\,\mathrm{K}^{-1}$), for homopolymers when changing a single C atom into O and S atoms in (a), (b), and (c), respectively ((poly[(diaminodiphenylmethane)-alt-(pyromellitic anhydride)], poly[(diaminodiphenylether)-alt-(pyromellitic anhydride)], poly[(diaminodiphenylsulfide)-alt-(pyromellitic anhydride)]). For SMILES-X top and bottom rows show 1D and 2D attention maps, respectively. For RF fingerprint similarity maps are shown. The darker the shade of the colour is, the stronger is the attention paid by the respective ensemble model. At the bottom of each figure, the experimental results are given as median with a range of available values, and the predictions are given as mean with one standard deviation.


*c1ccc(C)c(N2C(=O)c3ccc(-c4ccc5c(c4)C(=O)N(*)C5=O)cc3C2=O)c1
*c1ccc(N2C(=O)c3ccc(-c4ccc5c(c4)C(=O)N(*)C5=O)cc3C2=O)c(C)c1
*c1cccc(N2C(=O)c3ccc(-c4ccc5c(c4)C(=O)N(*)C5=O)cc3C2=O)c1C
O
N
*
Experimental: 3.93 $^{+0.74}_{-0.74}$ (10^{-5} K^{-1})
Predicted: 4.22 ± 0.64 (10^{-5} K^{-1})
Experimental: 1.52 $^{+0.94}_{-0.94}$ (10^{-5} K^{-1})
Predicted: 3.25 ± 0.54 (10^{-5} K^{-1})
Experimental: 4.12 $^{+0.12}_{-0.12}$ (10^{-5} K^{-1})
Predicted: 2.98 ± 0.81 (10^{-5} K^{-1})
Experimental: 3.93 $^{+0.74}_{-0.74}$
Predicted: 1.82 ± 0.10
Experimental: 1.51 $^{+0.94}_{-0.93}$
Predicted: 2.54 ± 0.50
Experimental: 4.12 $^{+0.12}_{-0.12}$
Predicted: 2.49 ± 0.37

(a) (b) (c)

FIGURE 5.7: Attention maps built upon the SMILES-X (red) and RF (blue) predictive ensemble models for the coefficient of linear thermal expansion, β ($10^{-5}\,\mathrm{K}^{-1}$), for homopolymers when changing the placement of a branch in (a) and (c), or the alignment of a wildcard with the main chain in (b) (poly[(4-methyl-m-phenylenediamine)-alt-(biphenyl-3,3':4,4'-tetracarboxylic dianhydride)], poly[(2-methyl-p-phenylenediamine)-alt-(biphenyl-3,3':4,4'-tetracarboxylic dianhydride)], poly[(2-methyl-m-phenylenediamine)-alt-(biphenyl-3,3':4,4'-tetracarboxylic dianhydride)]). For SMILES-X top and bottom rows show 1D and 2D attention maps, respectively. For RF fingerprint similarity maps are shown. The darker the shade of the colour is, the stronger is the attention paid by the respective ensemble model. At the bottom of each figure, the experimental results are given as median with a range of available values, and the predictions are given as mean with one standard deviation.

The prevailing importance of a repeating unit shape can be further confirmed when comparing homopolymers differing by a single atom: replacing a C atom in the poly[(diaminodiphenylmethane)-alt-(pyromellitic anhydride)] (Figure 5.6a) by an O (Figure 5.6b), or an S (Figure 5.6c) atom, shows no significant difference between the experimental values, but also between predictions and attention maps. Figures 5.7a, 5.7b, 5.7c demonstrate that to some extent the trained model distinguishes between the placements of a branch and the alignments of a wildcard with the main chain.

In this way, attention maps indicate that the final trained ensemble models make predictions on interpretable basis. The full list of attention maps for each homopolymer within our dataset can be found in Supplementary Materials, both for out-of-sample and in-sample cases.

5.6.3 Group contribution method

Here we compare the results of the prediction for the CTE to an expression derived from the GCM. In the original work van Krevelen build a model not for linear thermal expansion but for molar thermal expansion, denoted as E. Therefore, in order to compare our results, we needed to derive the formula for β instead.

Van Krevelen defines E as follows:

$$E = \left(\frac{\partial \mathbf{V}}{\partial T}\right)_p , \tag{5.3}$$

where $\mathbf{V}$ is the molar volume.

Observing the experimental data, he recognises a linear trend between van der Waals volumes $\mathrm{V_W}$, computed by GCMs, and the measured values of molar thermal expansion E, with different slope angles for glassy (E_g) and rubbery (or "liquid", E_l) states:

$$\begin{aligned} E_g &\simeq 0.45 \times 10^{-3}\,\mathrm{V_W}, \\ E_l &\simeq 1.00 \times 10^{-3}\,\mathrm{V_W}. \end{aligned} \tag{5.4}$$

Since our model is based on linear thermal expansion in glassy state, in order to compare the prediction results, we need to express it in terms of E_g. First, let us formally define the coefficient of volumetric expansion, α:

$$\alpha = \frac{1}{\mathrm{V}_{\mathrm{init}}} \left(\frac{\partial \mathrm{V}}{\partial T} \right)_p , \tag{5.5}$$

where $\mathrm{V}_{\mathrm{init}}$ is the initial volume of the material, $\partial\mathrm{V}$ is the change in volume given ∂T difference in temperatures at constant pressure p. Then, combining Equation 5.5 with definitions of β and E, in Equations 2.1 and 5.3, respectively:

$$\beta = \frac{1}{3}\alpha = \frac{1}{3}\frac{E}{\mathbf{V}_{init}}, \tag{5.6}$$

where the latter equation holds for stereoisomers.

The initial volume $\mathbf{V}_{init}$ depends on the experimental setup, specifically on the starting temperature. In case where the volume increases linearly with temperature (which is not always a good approximation), $\mathbf{V}_{init}$ can be expressed as:

$$\mathbf{V}_{init} \equiv \mathbf{V}(T_{init}) = \mathbf{V}_g(0) + E\,T_{init}, \tag{5.7}$$

where $\mathbf{V}_g(0)$ is the molar volume of a polymer at 0 K.

The value of $\mathbf{V}_g(0)$ is estimated by van Krevelen through a model based on the Simha and Boyer's concept of thermal expansion [90]. There, two important assumptions are made. The first one is that the molar thermal expansions are approximately equal between glassy and crystalline states. In other words,

$$\mathbf{V}_g(T_g) - \mathbf{V}_c(T_g) = \mathbf{V}_g(0) - \mathbf{V}_c(0). \tag{5.8}$$

The second assumption is that the hypothetical occupied volume of a polymer in crystalline state at 0 K, $\mathbf{V}_c(0)$, is equal to that of an undercooled liquid (which does not seem to be correct in general):

$$\mathbf{V}_l(0) = \mathbf{V}_c(0). \tag{5.9}$$

Equations 5.8, 5.9 allow to estimate $\mathrm{V_g}(0)$ as:

$$\mathbf{V}_g(0) = \mathbf{V}_c(0) + (E_l - E_g)\, T_g. \tag{5.10}$$

Remaining unknown $\mathbf{V}_c(0)$ is closely related to the $\mathbf{V}_W$, and van Krevelen gives the following approximation, referring to Bondi [91]:

$$\mathbf{V}_c(0) \simeq 1.3\, \mathrm{V_W}. \tag{5.11}$$

Substituting Equations 5.4, 5.7, 5.10 and 5.11 into Equation 5.6, the CLTE for polymers in glassy state can be approximated as:

$$\beta \simeq \frac{0.15 \times 10^{-3}}{1.3 + 0.55 \times 10^{-3}\, T_g + 0.45 \times 10^{-3}\, T_{init}}. \tag{5.12}$$

It is worth noting that if linear relations with van der Waals volume $\mathrm{V_W}$ in Equations 5.4 and 5.11 hold, $\mathrm{V_W}$ in the numerator and denominator cancel out. This means that while being developed based on the GCM, ultimately β does not depend on $\mathrm{V_W}$, and therefore is not directly related to the GCM.

Figure 5.8 demonstrates the prediction vs. observation performance for SMILES-X, RF and GCM models based on 18 data points for which experimental values of both $\mathrm{T_g}$ and $\mathrm{T_{init}}$ are known. It is obvious that the model based on van Krevelen's assumptions does not meet the experimental β values (predicted values are roughly constant). On the other hand, the ensemble machine learning models show satisfactory agreement. Thus, for linear thermal expansion the machine learning approach appears to be a relevant alternative to existing semi-empirical van Krevelen's model.

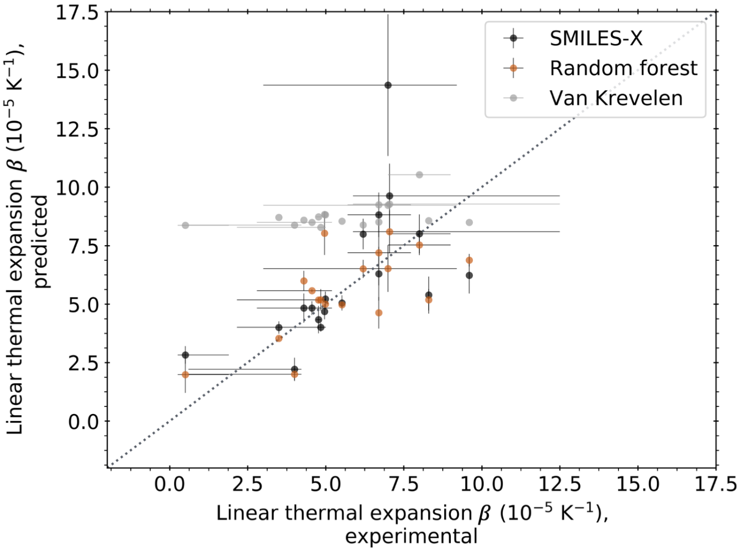


FIGURE 5.8: Comparison of the coefficient of linear thermal expansion, β ($10^{-5}\,\mathrm{K}^{-1}$), values computed using the formalism of van Krevelen (grey dots) against the SMILES-X (black dots) and RF (orange dots) ensemble model predictions. Error bars on the x-axis span between experimental minimum and maximum values, and error bars on the y-axis represent one standard deviation from the distribution of the predicted values.

5.7 Conclusions on Chapter 5

In this Chapter we present machine learning prediction models for the coefficient of linear thermal expansion, β ($10^{-5}\,\mathrm{K}^{-1}$). The main model is built with the SMILES-X molecular properties prediction software. We modified the original version to improve the hyperparameters search phase, implementing trainless NAS method developed in Chapter 4 of this thesis. The second auxiliary baseline model is built with RF.

The prediction for thermal expansivity, β, shows R^2-score of 68 %, which is a very good performance given humble 106 samples composing the dataset. RF model shows comparable fit to SMILES-X, with many polymers having similar prediction values between the two models. This is a very encouraging observation, as the two ML methods are of different nature, and such proximity of predictions increases credibility of both models.

We further validate the predictive model with experimental measurement, by analysing interpretation maps provided by the SMILES-X and by comparing with a classical semi-empirical model. For the experimental measurement, prediction meets the experiment with excellent agreement. Interpretation maps show hints that in a sense the model have developed some understanding of physicochemical mechanisms responsible for the polymer expansivity. Finally, the comparison with existing method unambiguously favours our SMILES-X machine learning model.

Following our results, we make the following conclusions:

- The trainless NAS method can likely be used with NLP architectures.
- It is possible to predict polymer thermal properties with NLP-based ML model. The quality of predictions depends on the data quality and quantity. Predictions prove reliable starting from about 100 samples in the dataset.
- It is possible to predict polymer thermal properties based solely on the repeating unit structure.
- Presented predictive model for the coefficient of linear thermal expansion of polymers, β can be used to replace experimental measurements.

This study demonstrates that NLP-based ML enables us to predict polymer properties which has been difficult because of polymers complex structure.

We acknowledge here that a part of the study in this chapter is published as [P3] in the list of publications.

6. Conclusions

This thesis aims to contribute to the acceleration of the development of novel polymers with machine learning (ML). For this we develop machine learning predictive models for the coefficient of linear thermal expansion, β. These models should be able to make property prediction for a polymer based directly on its chemical structure. We argue that due to the presence of long-range interactions within the molecular data, natural language processing (NLP) neural architectures should show the best performance for out problem. Therefore, we choose an NLP-based machine learning software, SMILES-X, for the models training.

In Chapter 4, we propose a novel method for neural architecture search (NAS) without training to optimise the SMILES-X architecture efficiently. All of the existing NAS methods require training. Therefore, hyperparameters related to the geometry optimisations should be found on simultaneously with training hyperparameters. This substantially limits the performance of existing NAS techniques. To address this issue, we develop a fully trainless NAS method. We explore relationships between the architecture prediction performance and some of its metrics prior to training. Specifically, we analyse the way different networks propagate the signals for different inputs and for different weight initialisations. We recognise that networks showing low variance over initialisations and high variance over the batch of data perform better. Based on this, we develop a trainless NAS metric $CV_u^{acc} = \sigma_u^{acc}/\mu_u^{acc}$. When tested on NAS-Bench-201 benchmark set of trained architectures [67], CV_u^{acc} achieves accuracies of 91.90 ± 2.27, 64.08 ± 5.63 and 38.76 ± 6.62 for CIFAR-10, CIFAR-100 [79] and a down-scaled version of ImageNet [80], respectively.

This success means that a good architecture should be stable against weight initialisations. This conclusion may have far going implications not only in the field of ML, but also in cognitive sciences, as it shows that learning and generalisation power of a system is predefined by the geometry of the connections.

The CV_u^{acc} metric significantly accelerates the NAS process, with a single architecture evaluation taking only 1.7 s (for a typical architecture within NAS-Bench-201 benchmark set with 8M parameters on average).

Finally, trainless NAS allows to differentiate training hyperparameters search, thus reducing the search space for whatever the algorithm that is used for their optimisation. In case of the SMILES-X, original software uses Bayesian optimisation (BO) to find the best combination of the number of units in the embedding, long short-term memory and dense layers together with batch size (BS) and learning rate (LT). In this scenario, BO operates in a 5-dimensional complex space. With implementation of the trainless NAS method, BO needs to optimise only BS and LT, by scanning a 2-dimensional space. Therefore, trainless NAS indirectly improves the efficiency of training hyperparameters search as well.

The presented trainless NAS algorithm is task-agnostic, as it does not make any assumptions on the network structure and is based on the neuron values of the last layer of a network. Its interpretation also seems to be general enough to be applicable to every field of ML. More work needs to be done to prove generalisability of the trainless scoring metric, both for SMILES-X and other types of neural architectures. Nevertheless, when implemented within SMILES-X, our trainless NAS points to stable networks with good convergence of learning curves.

In Chapter 5, we predicted one of the polymer thermal properties, the coefficient of linear thermal expansion, by using the trainless NAS method developed in Chapter 4. We present trained ensemble models which achieve the R^2-score of 68 %. Given the size and the quality of the data, this performance exceeds our expectations. We further validate the model with experimental measurement with lab-made poly(vinyl methyl ketone) samples. Ensemble model prediction shows excellent agreement. It is worth noting that the β value for poly(vinyl methyl ketone) lies in the region of high β values, which has relatively little data.

The attention mechanism implemented in SMILES-X allows to see which part of the SMILES string gets more attention from a model. It helps to compare model interpretation with existing domain knowledge. For example, it is known that polymers containing long carbon chains demonstrate high expansivities. On the other hand, polymers with rigid ring structures are known to have low β values. Attention maps of the β model often pays higher attention to these structures when doing predictions. This suggests that the machine learning model have developed comprehension of the mechanisms responsible for the thermal expansion of polymers.

The β model is particularly valuable, since there exist no structure-based model for polymer expansivity of sufficient precision. The only semi-empirical method shows almost constant predictions. Therefore, β model may significantly affect the development of polymers with lower coefficients of linear thermal expansion.

We also hope to improve the prediction results when more data becomes available. We conclude that for polymers a reasonable SMILES-X model can be built with a dataset of about a hundred samples.

Baseline RF models show comparable fit, many polymers having similar prediction values between the two models. This is a very encouraging observation, as the ML methods are of different nature, and such proximity of predictions increases credibility of both models.

To make the final conclusion, we have developed reliable predictive machine learning models for the coefficient of linear thermal expansion. This model can replace experiments and therefore to accelerate the development of polymers with better thermal properties.

The presented work provides many opportunities for future research. Thermal expansivity depends not only on the polymer chemical composition, but on the chain length as well. Meanwhile, the presented SMILES-X models are based only on the the structural information of a repeating unit. This limits the maximum predictive power of the SMILES-X. In future, we plan to modify the SMILES-X so that it takes descriptors such as chain length into account to further enhance the predictive models.

One can also build models for a wider range of critical properties, such as glass transition temperature or biodegradability. Ultimately, these models can be used for reversed problem of molecule generation, and incorporated into an active learning loop. This will help to achieve better data, better models and ultimately better polymers.

For NAS, we hope to verify the applicability of the developed trainless NAS to different machine learning problems and search spaces. It is also planned to test the performance of trainless scoring metric in combination with existing neural architecture generation algorithms.

Appendix

Fold	R^2-score	RMSE ($10^{-5}\,\mathrm{K}^{-1}$)	MAE ($10^{-5}\,\mathrm{K}^{-1}$)
	train/test	train/test	train/test
0	0.96 ± 0.01/0.94 ± 0.10	0.92 ± 0.09/0.49 ± 0.37	0.60 ± 0.05/0.36 ± 0.23
1	0.95 ± 0.01/0.43 ± 0.14	0.93 ± 0.13/2.96 ± 0.36	0.59 ± 0.05/2.28 ± 0.33
2	0.96 ± 0.01/0.79 ± 0.06	0.91 ± 0.11/1.71 ± 0.24	0.54 ± 0.05/0.85 ± 0.14
3	0.95 ± 0.01/0.15 ± 0.12	0.92 ± 0.11/2.78 ± 0.19	0.57 ± 0.05/2.55 ± 0.23
4	0.95 ± 0.01/0.21 ± 0.16	0.97 ± 0.09/3.45 ± 0.36	0.60 ± 0.05/2.36 ± 0.22
5	0.95 ± 0.01/0.56 ± 0.35	1.00 ± 0.09/0.75 ± 0.30	0.63 ± 0.05/0.48 ± 0.17
6	0.95 ± 0.01/ − 3.08 ± 0.69	0.97 ± 0.12/2.73 ± 0.23	0.60 ± 0.05/1.93 ± 0.20
7	0.96 ± 0.01/ − 0.15 ± 0.15	0.87 ± 0.11/4.63 ± 0.31	0.57 ± 0.05/2.78 ± 0.23
8	0.95 ± 0.01/0.90 ± 0.08	0.98 ± 0.10/1.31 ± 0.50	0.59 ± 0.05/1.06 ± 0.39
9	0.95 ± 0.01/0.21 ± 0.09	0.95 ± 0.10/4.52 ± 0.25	0.57 ± 0.05/3.50 ± 0.23
10	0.95 ± 0.01/0.47 ± 0.44	0.98 ± 0.11/1.44 ± 0.60	0.59 ± 0.05/0.88 ± 0.32
11	0.94 ± 0.01/0.84 ± 0.09	1.05 ± 0.10/2.04 ± 0.60	0.63 ± 0.05/1.55 ± 0.43
12	0.95 ± 0.01/0.95 ± 0.03	0.97 ± 0.09/0.90 ± 0.24	0.62 ± 0.05/0.79 ± 0.27
13	0.94 ± 0.01/0.67 ± 0.03	0.91 ± 0.11/5.04 ± 0.26	0.56 ± 0.05/3.75 ± 0.20
14	0.95 ± 0.01/0.15 ± 0.18	0.95 ± 0.10/2.72 ± 0.29	0.58 ± 0.05/1.85 ± 0.21
15	0.95 ± 0.01/0.95 ± 0.01	0.95 ± 0.12/1.18 ± 0.15	0.58 ± 0.05/0.98 ± 0.27
16	0.95 ± 0.01/0.68 ± 0.07	0.96 ± 0.09/1.75 ± 0.18	0.61 ± 0.04/1.23 ± 0.11
17	0.95 ± 0.01/0.45 ± 0.47	1.00 ± 0.12/0.68 ± 0.29	0.61 ± 0.05/0.56 ± 0.25
18	0.95 ± 0.01/0.85 ± 0.06	0.97 ± 0.09/0.86 ± 0.18	0.59 ± 0.05/0.67 ± 0.24
19	0.94 ± 0.01/0.92 ± 0.05	1.00 ± 0.10/1.44 ± 0.47	0.64 ± 0.05/1.03 ± 0.28

TABLE 6.1: R^2-score, RMSE and MAE of the trained RF ensemble model on the coefficient of linear thermal expansion, β ($10^{-5}\,\mathrm{K}^{-1}$), fold-wise on the training and test sets. The reported mean is calculated over 5 trained models, each trained with a different random number seed. The error corresponds to a single standard deviation.

Fold	R^2-score	RMSE ($10^{-5}\,\mathrm{K}^{-1}$)	MAE ($10^{-5}\,\mathrm{K}^{-1}$)
	train/test	train/test	train/test
0	$0.99 \pm 0.00/0.78 \pm 0.07$	$0.41 \pm 0.11/0.89 \pm 0.15$	$0.30 \pm 0.08/0.80 \pm 0.13$
1	$1.00 \pm 0.00/0.53 \pm 0.33$	$0.28 \pm 0.06/2.54 \pm 0.86$	$0.19 \pm 0.03/1.98 \pm 0.48$
2	$0.99 \pm 0.01/0.84 \pm 0.06$	$0.44 \pm 0.16/1.43 \pm 0.26$	$0.26 \pm 0.06/0.86 \pm 0.06$
3	$0.99 \pm 0.01/-0.37 \pm 0.84$	$0.43 \pm 0.12/3.31 \pm 1.21$	$0.30 \pm 0.08/2.63 \pm 0.94$
4	$0.99 \pm 0.01/0.83 \pm 0.04$	$0.29 \pm 0.15/1.58 \pm 0.20$	$0.20 \pm 0.09/1.28 \pm 0.17$
5	$0.99 \pm 0.00/0.62 \pm 0.13$	$0.35 \pm 0.11/0.69 \pm 0.12$	$0.24 \pm 0.07/0.54 \pm 0.10$
6	$1.00 \pm 0.00/-3.49 \pm 0.82$	$0.23 \pm 0.06/2.85 \pm 0.26$	$0.17 \pm 0.04/2.03 \pm 0.26$
7	$0.99 \pm 0.00/-0.43 \pm 0.13$	$0.32 \pm 0.11/5.17 \pm 0.23$	$0.25 \pm 0.08/3.53 \pm 0.30$
8	$0.99 \pm 0.01/0.66 \pm 0.17$	$0.38 \pm 0.14/2.28 \pm 0.62$	$0.29 \pm 0.10/1.67 \pm 0.42$
9	$0.99 \pm 0.01/0.26 \pm 0.16$	$0.39 \pm 0.16/4.37 \pm 0.48$	$0.24 \pm 0.08/2.95 \pm 0.34$
10	$0.99 \pm 0.01/0.13 \pm 0.24$	$0.39 \pm 0.16/1.82 \pm 0.26$	$0.25 \pm 0.09/1.32 \pm 0.25$
11	$0.99 \pm 0.01/0.18 \pm 0.48$	$0.42 \pm 0.20/4.43 \pm 1.34$	$0.30 \pm 0.12/3.38 \pm 1.05$
12	$0.99 \pm 0.01/0.88 \pm 0.06$	$0.47 \pm 0.19/1.33 \pm 0.32$	$0.31 \pm 0.11/1.16 \pm 0.27$
13	$0.99 \pm 0.00/0.83 \pm 0.02$	$0.30 \pm 0.09/3.64 \pm 0.17$	$0.23 \pm 0.07/2.66 \pm 0.26$
14	$1.00 \pm 0.00/0.50 \pm 0.13$	$0.29 \pm 0.10/2.08 \pm 0.26$	$0.21 \pm 0.05/1.34 \pm 0.22$
15	$1.00 \pm 0.00/0.94 \pm 0.02$	$0.21 \pm 0.10/1.28 \pm 0.30$	$0.15 \pm 0.06/1.03 \pm 0.24$
16	$0.99 \pm 0.00/0.68 \pm 0.11$	$0.28 \pm 0.10/1.72 \pm 0.27$	$0.18 \pm 0.05/1.28 \pm 0.22$
17	$1.00 \pm 0.00/-1.46 \pm 1.50$	$0.25 \pm 0.11/1.37 \pm 0.47$	$0.17 \pm 0.07/1.07 \pm 0.35$
18	$1.00 \pm 0.00/0.71 \pm 0.09$	$0.30 \pm 0.08/1.20 \pm 0.20$	$0.23 \pm 0.07/0.96 \pm 0.19$
19	$0.99 \pm 0.00/0.95 \pm 0.02$	$0.28 \pm 0.12/1.10 \pm 0.28$	$0.16 \pm 0.04/0.83 \pm 0.22$

Table 6.2: R^2-score, RMSE and MAE of the trained SMILES-X ensemble model on the coefficient of linear thermal expansion, β ($10^{-5}\,\mathrm{K}^{-1}$), fold-wise on the training and test sets. The reported mean is calculated over 5 trained models, each trained with a different random number seed. The error corresponds to a single standard deviation.

6.1 Poly(vinyl methyl ketone) measurement details

Details on the coefficient of linear thermal expansion, β, measurement for poly(vinyl methyl ketone) are summarised in the Table 6.4. Temperature curves for each of the 4 samples are given on Figure 6.5.

Table 6.3: Performance of the CV_u^{acc} trainless NAS metric on CIFAR-10, CIFAR-100 and ImageNet16-120 with different number of architectures and varying batch size.

CIFAR-100

		Batch size									
		2		4		8		16		32	
N_a	N_{init}	Validation	Test	Validation	Test	Validation	Test	Validation	Test	Validation	Test
10	10	59.44 ± 12.50	59.54 ± 12.51	65.71 ± 6.03	65.86 ± 6.02	62.71 ± 8.44	62.84 ± 8.44	62.00 ± 8.75	62.09 ± 8.82	62.46 ± 8.16	62.58 ± 8.19
	100	62.30 ± 9.05	62.44 ± 9.07	62.62 ± 7.28	62.71 ± 7.27	62.33 ± 7.58	62.44 ± 7.57	63.65 ± 6.93	63.71 ± 6.99	59.60 ± 8.95	59.71 ± 9.04
25	10	58.00 ± 13.99	58.11 ± 13.95	66.33 ± 4.84	66.48 ± 4.84	65.67 ± 5.22	65.79 ± 5.22	64.60 ± 5.23	64.67 ± 5.30	60.45 ± 9.01	60.52 ± 9.08
	100	60.14 ± 10.26	60.23 ± 10.24	62.38 ± 7.46	62.48 ± 7.49	62.47 ± 5.97	62.57 ± 6.00	63.76 ± 6.84	63.84 ± 6.91	58.77 ± 9.31	58.86 ± 9.37
100	10	58.06 ± 13.65	58.19 ± 13.64	66.64 ± 3.24	66.74 ± 3.27	66.20 ± 4.02	66.27 ± 4.05	64.80 ± 4.85	64.94 ± 4.94	59.60 ± 8.95	59.71 ± 9.04
	100	60.42 ± 8.19	60.49 ± 8.18	63.30 ± 6.11	63.38 ± 6.09	61.87 ± 5.83	61.98 ± 5.87	65.78 ± 5.29	65.88 ± 5.32	60.08 ± 8.62	60.15 ± 8.71

		Batch size							
		64		128		256		512	
N_a	N_{init}	Validation	Test	Validation	Test	Validation	Test	Validation	Test
10	10	62.20 ± 7.88	62.33 ± 7.89	63.21 ± 7.11	63.31 ± 7.19	62.96 ± 7.68	63.05 ± 7.67	62.20 ± 7.97	62.32 ± 8.00
	100	59.77 ± 10.35	59.88 ± 10.38	60.46 ± 8.83	60.59 ± 8.86	63.73 ± 5.62	63.83 ± 5.65	64.28 ± 5.45	64.38 ± 5.49
25	10	62.49 ± 7.37	62.61 ± 7.40	63.42 ± 7.02	63.53 ± 7.08	61.91 ± 8.13	62.05 ± 8.14	61.51 ± 8.48	61.63 ± 8.51
	100	60.03 ± 9.68	60.15 ± 9.68	60.34 ± 7.97	60.45 ± 8.00	63.83 ± 5.50	63.92 ± 5.57	64.15 ± 5.11	64.23 ± 5.19
100	10	63.10 ± 6.03	63.21 ± 6.01	63.74 ± 5.93	63.89 ± 5.97	61.98 ± 7.44	62.09 ± 7.45	60.82 ± 8.81	60.96 ± 8.82
	100	60.54 ± 8.54	60.64 ± 8.56	60.91 ± 7.84	61.01 ± 7.87	63.99 ± 5.61	64.08 ± 5.63	64.01 ± 5.06	64.10 ± 5.10

CIFAR-10

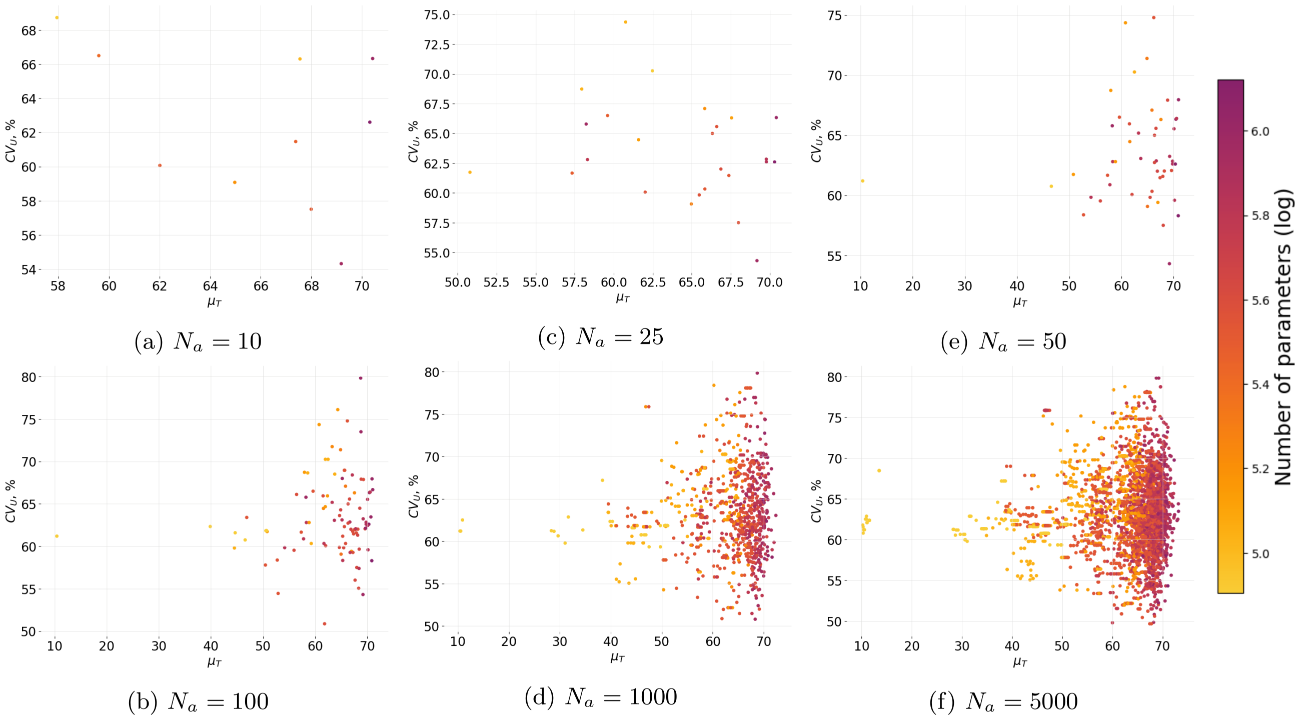
		Batch size									
		2		4		8		16		32	
N_a	N_{init}	Validation	Test	Validation	Test	Validation	Test	Validation	Test	Validation	Test
10	10	84.02 ± 7.41	88.45 ± 5.70	85.90 ± 4.04	89.03 ± 4.75	85.28 ± 5.78	90.79 ± 3.21	86.62 ± 4.00	88.67 ± 7.46	83.20 ± 7.18	87.52 ± 8.04
	100	83.94 ± 6.10	90.68 ± 2.94	85.79 ± 5.33	91.50 ± 2.71	83.32 ± 8.00	91.46 ± 2.42	85.01 ± 6.08	90.96 ± 3.07	85.63 ± 5.05	90.55 ± 3.24
25	10	84.14 ± 6.68	88.42 ± 4.93	85.98 ± 4.11	89.37 ± 4.16	85.42 ± 6.16	91.08 ± 3.38	86.49 ± 4.15	89.89 ± 5.49	83.04 ± 6.37	87.62 ± 7.68
	100	88.42 ± 4.93	90.79 ± 3.51	89.37 ± 4.16	91.82 ± 2.55	91.08 ± 3.38	91.56 ± 1.74	89.89 ± 5.49	90.89 ± 2.96	87.62 ± 7.68	91.15 ± 2.29
100	10	84.80 ± 4.52	87.99 ± 5.02	85.51 ± 3.72	89.85 ± 3.55	86.23 ± 5.05	91.57 ± 1.77	86.37 ± 3.59	90.49 ± 4.09	82.02 ± 6.49	87.99 ± 8.17
	100	83.81 ± 5.20	91.04 ± 2.50	87.04 ± 3.99	92.37 ± 1.85	82.68 ± 7.62	91.57 ± 1.65	84.89 ± 6.39	91.43 ± 1.64	86.23 ± 4.13	91.36 ± 1.96

		Batch size							
		64		128		256		512	
N_a	N_{init}	Validation	Test	Validation	Test	Validation	Test	Validation	Test
10	10	84.65 ± 6.66	87.31 ± 7.86	85.31 ± 5.42	88.65 ± 7.35	85.60 ± 5.02	89.97 ± 3.89	86.35 ± 4.25	91.15 ± 2.48
	100	86.19 ± 4.94	91.19 ± 2.71	87.59 ± 2.55	90.49 ± 3.67	87.98 ± 1.95	91.03 ± 2.77	88.29 ± 1.64	91.48 ± 1.82
25	10	83.82 ± 6.81	87.31 ± 7.54	85.71 ± 4.72	87.98 ± 8.87	85.48 ± 5.36	90.13 ± 3.59	86.68 ± 2.95	91.34 ± 1.93
	100	86.29 ± 3.60	91.75 ± 2.17	87.63 ± 2.21	91.14 ± 3.22	88.03 ± 1.74	91.46 ± 2.39	88.27 ± 1.48	91.51 ± 1.75
100	10	83.80 ± 6.39	88.18 ± 7.12	86.27 ± 3.97	88.88 ± 7.90	86.05 ± 4.56	90.33 ± 3.86	86.44 ± 2.57	91.25 ± 2.23
	100	86.39 ± 3.31	92.50 ± 1.59	87.49 ± 2.46	92.32 ± 2.16	88.18 ± 1.66	91.90 ± 2.27	88.39 ± 1.37	91.52 ± 1.87

ImageNet16-120

		Batch size									
		2		4		8		16		32	
N_a	N_{init}	Validation	Test	Validation	Test	Validation	Test	Validation	Test	Validation	Test
10	10	31.40 ± 8.43	30.96 ± 8.66	33.02 ± 7.71	31.85 ± 8.62	33.73 ± 7.58	33.39 ± 7.83	34.29 ± 6.87	34.04 ± 7.15	32.78 ± 7.51	32.37 ± 7.83
	100	31.97 ± 7.78	31.49 ± 8.02	37.46 ± 6.53	37.39 ± 6.78	37.96 ± 6.08	37.91 ± 6.33	37.74 ± 6.89	37.70 ± 7.13	35.72 ± 9.34	35.59 ± 9.69
25	10	32.09 ± 8.11	31.66 ± 8.39	32.31 ± 8.36	31.93 ± 8.66	34.95 ± 6.78	34.71 ± 7.03	34.11 ± 6.50	33.84 ± 6.78	32.93 ± 6.95	32.53 ± 7.23
	100	36.67 ± 6.63	36.59 ± 6.93	37.86 ± 5.93	37.82 ± 6.21	38.44 ± 5.90	38.47 ± 6.19	38.42 ± 6.30	38.42 ± 6.51	36.44 ± 9.49	36.37 ± 9.84
100	10	31.97 ± 7.78	31.49 ± 8.02	32.66 ± 8.48	32.36 ± 8.75	34.13 ± 6.89	33.81 ± 7.11	34.99 ± 5.39	34.69 ± 5.66	32.97 ± 5.94	32.46 ± 6.19
	100	36.93 ± 6.38	36.89 ± 6.64	38.78 ± 5.60	38.72 ± 5.85	39.11 ± 5.03	39.17 ± 5.23	38.68 ± 6.34	38.76 ± 6.62	36.73 ± 9.92	36.73 ± 10.23

		Batch size							
		64		128		256		512	
N_a	N_{init}	Validation	Test	Validation	Test	Validation	Test	Validation	Test
10	10	33.33 ± 7.40	33.02 ± 7.71	32.54 ± 8.11	32.85 ± 7.78	34.02 ± 7.91	33.72 ± 8.30	33.42 ± 7.61	33.11 ± 7.94
	100	36.55 ± 8.16	36.47 ± 8.49	36.21 ± 8.12	36.09 ± 8.43	35.64 ± 8.42	35.50 ± 8.77	35.04 ± 8.40	34.90 ± 8.72
25	10	33.39 ± 6.87	33.02 ± 7.11	32.32 ± 7.95	32.66 ± 7.65	33.90 ± 7.74	33.58 ± 8.08	33.37 ± 7.65	33.06 ± 7.99
	100	37.77 ± 7.76	37.75 ± 8.03	35.64 ± 8.82	35.51 ± 9.13	36.75 ± 7.84	36.71 ± 8.14	34.92 ± 8.18	34.74 ± 8.45
100	10	34.02 ± 6.41	33.64 ± 6.64	32.69 ± 7.06	33.06 ± 6.81	34.08 ± 7.39	33.79 ± 7.73	34.04 ± 7.12	33.79 ± 7.49
	100	37.48 ± 8.54	37.49 ± 8.83	37.35 ± 7.69	37.27 ± 8.05	38.11 ± 6.75	38.11 ± 7.06	35.61 ± 7.81	35.46 ± 8.08



(a) $N_a = 10$

(c) $N_a = 25$

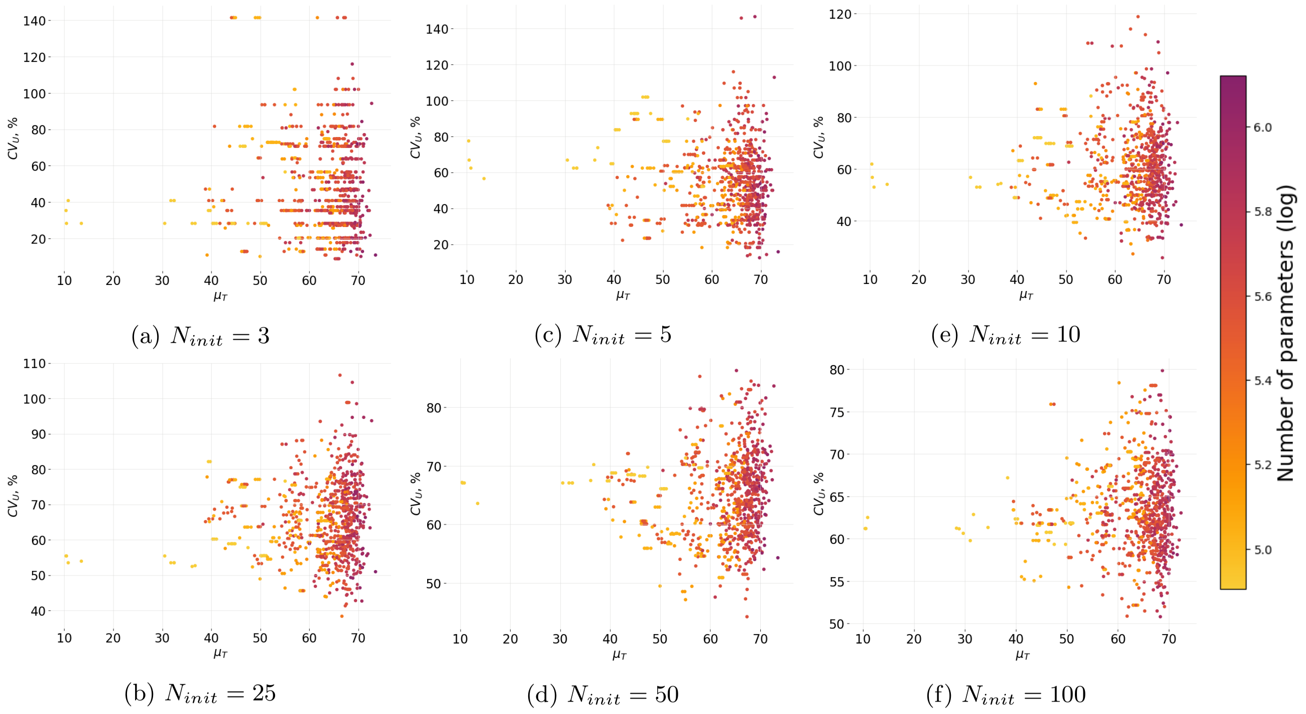
(e) $N_a = 50$

(b) $N_a = 100$

(d) $N_a = 1000$

(f) $N_a = 5000$

FIGURE 6.1: Comparison of the relative standard deviation CV_U (%) performance against mean trained accuracy μ_T for CIFAR-100 dataset for different number of selected architectures $N_a \in [10, 25, 50, 100, 1000, 5000]$. Statistics are computed over $N_{init} = 100$ initialisations. One point represents one architecture. The colours represent the logarithm of the total number of trained parameters.



(a) $N_{init} = 3$

(c) $N_{init} = 5$

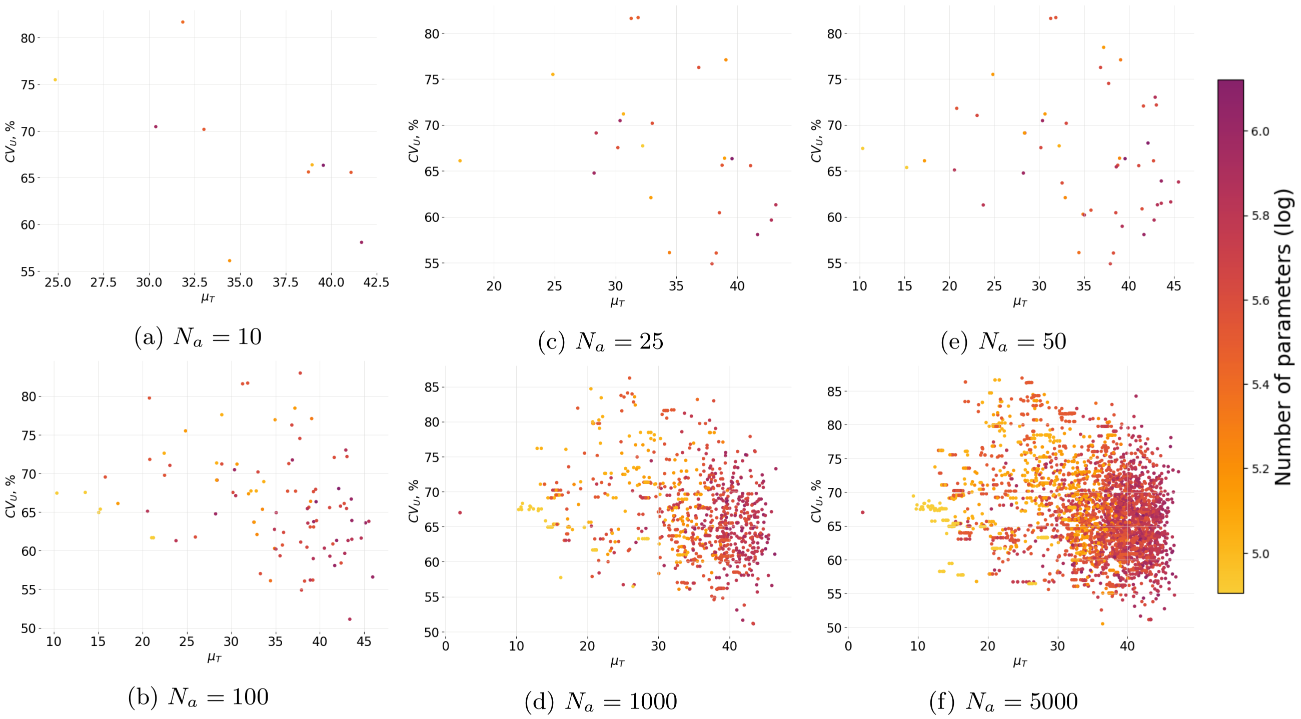
(e) $N_{init} = 10$

(b) $N_{init} = 25$

(d) $N_{init} = 50$

(f) $N_{init} = 100$

FIGURE 6.2: Comparison of the relative standard deviation CV_U (%) performance against mean trained accuracy μ_T for CIFAR-100 dataset. Statistics are computed over varying number of initialisations $N_{init} \in [3, 5, 10, 25, 50, 100]$. One point stands for one architecture. The colours represent the logarithm of the total number of trained parameters.



(a) $N_a = 10$

(c) $N_a = 25$

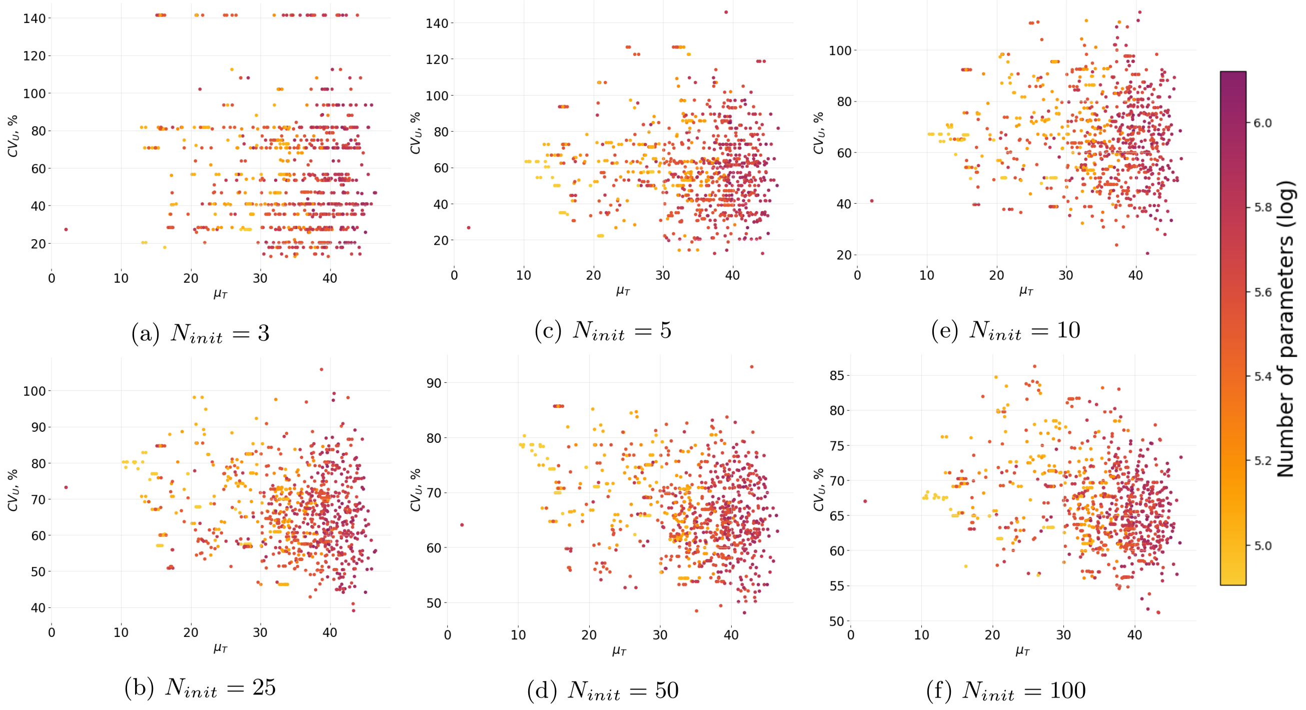
(e) $N_a = 50$

(b) $N_a = 100$

(d) $N_a = 1000$

(f) $N_a = 5000$

FIGURE 6.3: Comparison of the relative standard deviation CV_U (%) performance against mean trained accuracy μ_T for ImageNet16-120 dataset for different number of selected architectures $N_a \in [10, 25, 50, 100, 1000, 5000]$. Statistics are computed over $N_{init} = 100$ initialisations. One point represents one architecture. The colours represent the logarithm of the total number of trained parameters.



(a) $N_{init} = 3$

(c) $N_{init} = 5$

(e) $N_{init} = 10$

(b) $N_{init} = 25$

(d) $N_{init} = 50$

(f) $N_{init} = 100$

FIGURE 6.4: Comparison of the relative standard deviation CV_U (%) performance against mean trained accuracy μ_T for ImageNet16-120 dataset. Statistics are computed over varying number of initialisations $N_{init} \in [3, 5, 10, 25, 50, 100]$. One point stands for one architecture. The colours represent the logarithm of the total number of trained parameters.

TABLE 6.4: Measurement of the coefficient of linear thermal expansion, β, for poly(vinyl methyl ketone). Experimental temperature range corresponds to the full range set during the experiment. Calculation range is the range below the glass transition temperature used for the calculation of the β.

Method	Thermomechanical analysis (TMA)
Applied force	49.0 mN
Heating rate	5 °C/min
Measurement rate	1/s
Atmosphere	N_2
Experimental range (full)	20 – 60 °C
Calculation range (used)	20 – 30 °C

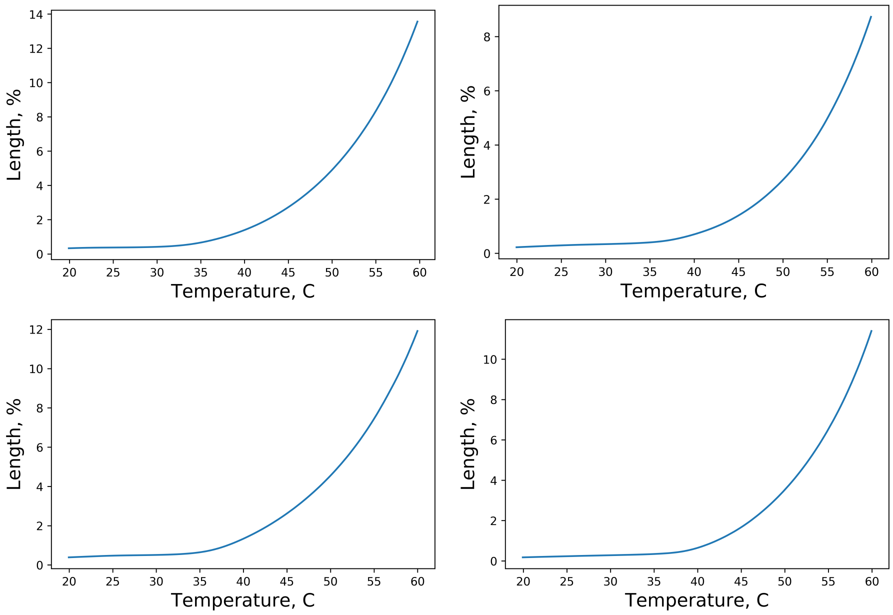


FIGURE 6.5: Temperature curves for each of 4 samples of poly(vinyl methyl ketone).

Index

Acronyms

BO Bayesian Optimisation. 22, 23, 47–50, 71

BPDA Biphenyltetracarboxylic Dianhydride. v, vi, 59, 60, 62

BS Batch Size. 27, 30, 31, 45, 47–50, 71

CNN Convolutional Neural Network. 18

CTE Coefficient of Thermal Expansion. 8–10, 12, 52, 55, 57–59, 65

GCM Group contribution method. 10, 11, 65

GNN Graph Neural Network. 18–20

LSTM Long Short-Term Memory. 19–21, 23, 24, 26, 28, 47, 50

LT Learning Rate. 30, 31, 47–50, 71

MAE Mean Absolute Error. 53, 74, 75

ML Machine Learning. 2, 15, 18–20, 30, 69–71

NAS Neural architecture search. vii, 2, 26–30, 35, 39, 40, 44–50, 68–73, 76

NLP Natural Language Processing. 18–23, 50, 69, 70

NN Neural Network. 18, 20, 23, 27–31, 46

PMDA Pyromellitic Dianhydride. v, vi, 59, 60, 62

RF Random Forest. v–vii, 16, 18, 21, 50, 51, 53–55, 57–59, 61–64, 67, 68, 72, 74

RMSE Root-Mean-Square Error. vii, 22, 48, 49, 53, 55, 74, 75

SMILES Simplified Molecular Input Line Entry System. i, 17–25, 50, 53, 55, 58

Bibliography

[1] J.-P. Correa-Baena, K. Hippalgaonkar, J. van Duren, *et al.*, "Accelerating materials development via automation, machine learning, and high-performance computing," *Joule*, vol. 2, no. 8, pp. 1410–1420, 2018.

[2] Z. Wang, G. W. Walker, D. C. Muir, and K. Nagatani-Yoshida, "Toward a global understanding of chemical pollution: A first comprehensive analysis of national and regional chemical inventories," *Environmental Science & Technology*, vol. 54, no. 5, pp. 2575–2584, 2020.

[3] N. B. Shenogina, M. Tsige, S. S. Patnaik, and S. M. Mukhopadhyay, "Molecular modeling approach to prediction of thermo-mechanical behavior of thermoset polymer networks," *Macromolecules*, vol. 45, no. 12, pp. 5307–5315, 2012.

[4] S. Pruksawan, G. Lambard, S. Samitsu, K. Sodeyama, and M. Naito, "Prediction and optimization of epoxy adhesive strength from a small dataset through active learning," *Science and technology of advanced materials*, vol. 20, no. 1, pp. 1010–1021, 2019.

[5] Y. Zhang and X. Xu, "Machine learning glass transition temperature of polymers," *Heliyon*, vol. 6, no. 10, e05055, 2020.

[6] J. Pei, C. Cai, X. Zhu, G. Wang, and B Yan, "Modeling the glass transition temperature of polymers via multipole moments using support vector regression," in *Advanced Materials Research*, Trans Tech Publ, vol. 455, 2012, pp. 430–435.

[7] X. Yu, B. Yi, X. Wang, and Z. Xie, "Correlation between the glass transition temperatures and multipole moments for polymers," *Chemical Physics*, vol. 332, no. 1, pp. 115–118, 2007.

[8] A. Mannodi-Kanakkithodi, G. Pilania, T. D. Huan, T. Lookman, and R. Ramprasad, "Machine learning strategy for accelerated design of polymer dielectrics," *Scientific reports*, vol. 6, no. 1, pp. 1–10, 2016.

[9] H Staudinger, *Berichte der deutschen chemischen Gessellschaft (A and B Series)*, vol. 53, pp. 1073–1085, 2006. DOI: 10.1002/cber.19200530627.

[10] H. F. Mark, *Encyclopedia of Polymer Science and Technology: Phenolic resins to Polyelectrolytes*. Interscience Publishers, 1969, vol. 10.

[11] W. B. Jensen, “The origin of the polymer concept,” *Journal of Chemical Education*, vol. 85, no. 5, p. 624, 2008.

[12] A. D. McNaught and A. Wilkinson. Blackwell Scientific Publications, 1997, ISBN: 0-9678550-9-8. DOI: 10.1351/goldbook.

[13] D. Hosler, S. L. Burkett, and M. J. Tarkanian, “Prehistoric polymers: Rubber processing in ancient mesoamerica,” *Science*, vol. 284, no. 5422, pp. 1988–1991, 1999.

[14] A. Rudin and P. Choi, *The elements of polymer science and engineering*. Academic press, 2012.

[15] D. W. Van Krevelen and K. Te Nijenhuis, *Properties of polymers: their correlation with chemical structure; their numerical estimation and prediction from additive group contributions*. Elsevier, 2009.

[16] A. A. Askadskii, *Computational materials science of polymers*. Cambridge Int Science Publishing, 2003.

[17] J. Bicerano, *Prediction of polymer properties*. cRc Press, 2002.

[18] V. Moruzzi, J. Janak, and K Schwarz, “Calculated thermal properties of metals,” *Physical Review B*, vol. 37, no. 2, p. 790, 1988.

[19] A. Ruffa, “Thermal expansion in insulating materials,” *Journal of Materials Science*, vol. 15, no. 9, pp. 2258–2267, 1980.

[20] A. Vashisth, C. Ashraf, C. E. Bakis, and A. C. van Duin, “Effect of chemical structure on thermo-mechanical properties of epoxy polymers: Comparison of accelerated reaxff simulations and experiments,” *Polymer*, vol. 158, pp. 354–363, 2018, ISSN: 0032-3861. DOI: https://doi.org/10.1016/j.polymer.2018.11.005.

[21] R. Simha and P. S. Wilson, “Thermal expansion of amorphous polymers at atmospheric pressure. ii. theoretical considerations,” *Macromolecules*, vol. 6, no. 6, pp. 908–914, 1973.

[22] J. Kardos, J Raisoni, S Piccarolo, and J. Halpin, “Prediction and measurement of the thermal expansion coefficient of crystalline polymers,” *Polymer Engineering & Science*, vol. 19, no. 14, pp. 1000–1009, 1979.

[23] R. A. Schapery, “Thermal expansion coefficients of composite materials based on energy principles,” *Journal of Composite Materials*, vol. 2, no. 3, pp. 380–404, 1968.

[24] Z. Kolská, J. Kukal, M. Zábranskỳ, and V. Růžička, “Estimation of the heat capacity of organic liquids as a function of temperature by a three-level group contribution method,” *Industrial & engineering chemistry research*, vol. 47, no. 6, pp. 2075–2085, 2008.

[25] D. C. Elton, Z. Boukouvalas, M. S. Butrico, M. D. Fuge, and P. W. Chung, “Applying machine learning techniques to predict the properties of energetic materials,” *Scientific reports*, vol. 8, no. 1, pp. 1–12, 2018.

[26] D. Weininger, “Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules,” *Journal of chemical information and computer sciences*, vol. 28, no. 1, pp. 31–36, 1988.

[27] A. O. Oliynyk, E. Antono, T. D. Sparks, *et al.*, “High-throughput machine-learning-driven synthesis of full-heusler compounds,” *Chemistry of Materials*, vol. 28, no. 20, pp. 7324–7331, 2016.

[28] D. S. Palmer, N. M. O’Boyle, R. C. Glen, and J. B. Mitchell, “Random forest models to predict aqueous solubility,” *Journal of chemical information and modeling*, vol. 47, no. 1, pp. 150–158, 2007.

[29] V. Svetnik, A. Liaw, C. Tong, J. C. Culberson, R. P. Sheridan, and B. P. Feuston, “Random forest: A classification and regression tool for compound classification and qsar modeling,” *Journal of chemical information and computer sciences*, vol. 43, no. 6, pp. 1947–1958, 2003.

[30] F. Legrain, J. Carrete, A. van Roekeghem, S. Curtarolo, and N. Mingo, “How chemical composition alone can predict vibrational free energies and entropies of solids,” *Chemistry of Materials*, vol. 29, no. 15, pp. 6220–6227, 2017.

[31] L. Breiman, “Random forests,” *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[32] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, “The graph neural network model,” *IEEE transactions on neural networks*, vol. 20, no. 1, pp. 61–80, 2008.

[33] M. Gori, G. Monfardini, and F. Scarselli, “A new model for learning in graph domains,” in *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, IEEE, vol. 2, 2005, pp. 729–734.

[34] J. M. Stokes, K. Yang, K. Swanson, *et al.*, “A deep learning approach to antibiotic discovery,” *Cell*, vol. 180, no. 4, pp. 688–702, 2020.

[35] O. Lange and L. Perez, *Traffic prediction with advanced graph neural networks*, 2020.

[36] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, “Attention is all you need,” *arXiv preprint arXiv:1706.03762*, 2017.

[37] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, “Graph attention networks,” *arXiv preprint arXiv:1710.10903*, 2017.

[38] Z. Xiong, D. Wang, X. Liu, *et al.*, “Pushing the boundaries of molecular representation for drug discovery with the graph attention mechanism,” *Journal of medicinal chemistry*, vol. 63, no. 16, pp. 8749–8760, 2019.

[39] C. Shang, Q. Liu, Q. Tong, J. Sun, M. Song, and J. Bi, “Multi-view spectral graph convolution with consistent edge attention for molecular modeling,” *Neurocomputing*, vol. 445, pp. 12–25, 2021.

[40] T. Xie, A. France-Lanord, Y. Wang, Y. Shao-Horn, and J. C. Grossman, “Graph dynamical networks for unsupervised learning of atomic scale dynamics in materials,” *Nature communications*, vol. 10, no. 1, pp. 1–9, 2019.

[41] U. Alon and E. Yahav, "On the bottleneck of graph neural networks and its practical implications," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=i80OPhOCVH2.

[42] C. Raffel and D. P. W. Ellis, *Feed-forward networks with attention can solve some long-term memory problems*, 2015. eprint: arXiv:1512.08756.

[43] D. Flam-Shepherd, K. Zhu, and A. Aspuru-Guzik, "Keeping it simple: Language models can learn complex molecular distributions," *arXiv preprint arXiv:2112.03041*, 2021.

[44] *GPyOpt: A Bayesian optimization framework in Python.* [Online]. Available: https:://github.com/SheffieldML/GPyOpt.

[45] M. Honnibal, *Embed, encode, attend, predict: The new deep learning formula for state-of-the-art NLP models.* [Online]. Available: https://explosion.ai/blog/deep-learning-formula-nlp.

[46] Z. Wu, B. Ramsundar, E. Feinberg, *et al.*, "MoleculeNet: A benchmark for molecular machine learning," *Chem. Sci.*, vol. 9, pp. 513–530, 2 2018. DOI: 10.1039/C7SC02664A.

[47] E. Real, S. Moore, A. Selle, *et al.*, "Large-scale evolution of image classifiers," *arXiv preprint arXiv:1703.01041*, 2017.

[48] H. Liu, K. Simonyan, O. Vinyals, C. Fernando, and K. Kavukcuoglu, "Hierarchical representations for efficient architecture search," *arXiv preprint arXiv:1711.00436*, 2017.

[49] M. Suganuma, S. Shirakawa, and T. Nagao, "A genetic programming approach to designing convolutional neural network architectures," in *Proceedings of the genetic and evolutionary computation conference*, 2017, pp. 497–504.

[50] T. Elsken, J. H. Metzen, and F. Hutter, "Efficient multi-objective neural architecture search via lamarckian evolution," *arXiv preprint arXiv:1804.09081*, 2018.

[51] P. J. Angeline, G. M. Saunders, and J. B. Pollack, "An evolutionary algorithm that constructs recurrent neural networks," *IEEE Transactions on Neural Networks*, vol. 5, no. 1, pp. 54–65, 1994.

[52] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le, "Regularized evolution for image classifier architecture search," in *Proceedings of the aaai conference on artificial intelligence*, vol. 33, 2019, pp. 4780–4789.

[53] P. I. Frazier, "A tutorial on bayesian optimization," *arXiv preprint arXiv:1807.02811*, 2018.

[54] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. De Freitas, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2015.

[55] J. Bergstra, D. Yamins, and D. D. Cox, "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures," in *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ser. ICML'13, Atlanta, GA, USA: JMLR.org, 2013, I–115–I–123.

[56] T. Domhan, J. T. Springenberg, and F. Hutter, "Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves," in *Proceedings of the 24th International Conference on Artificial Intelligence*, ser. IJCAI'15, Buenos Aires, Argentina: AAAI Press, 2015, 3460–3468, ISBN: 9781577357384.

[57] H. Mendoza, A. Klein, M. Feurer, J. T. Springenberg, and F. Hutter, "Towards automatically-tuned neural networks," in *Workshop on Automatic Machine Learning*, 2016, pp. 58–65.

[58] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," *arXiv preprint arXiv:1611.01578*, 2016.

[59] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3-4, pp. 229–256, 1992.

[60] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8697–8710.

[61] H. Pham, M. Y. Guan, B. Zoph, Q. V. Le, and J. Dean, "Efficient neural architecture search via parameter sharing," *arXiv preprint arXiv:1802.03268*, 2018.

[62] T Elsken, J. Metzen, and F Hutter, "Neural architecture search: A survey. arxiv 2018," *arXiv preprint arXiv:1808.05377*, 2018.

[63] C. G. Northcutt, A. Athalye, and J. Mueller, "Pervasive label errors in test sets destabilize machine learning benchmarks," *arXiv preprint arXiv:2103.14749*, 2021.

[64] R. Istrate, F. Scheidegger, G. Mariani, D. Nikolopoulos, C. Bekas, and A. C. I. Malossi, "Tapas: Train-less accuracy predictor for architecture search," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 3927–3934.

[65] B. Deng, J. Yan, and D. Lin, "Peephole: Predicting network performance before training," *arXiv preprint arXiv:1712.03351*, 2017.

[66] J. Mellor, J. Turner, A. Storkey, and E. J. Crowley, "Neural architecture search without training," *arXiv preprint arXiv:2006.04647v1*, 2020.

[67] X. Dong and Y. Yang, "Nas-bench-102: Extending the scope of reproducible neural architecture search," *arXiv preprint arXiv:2001.00326*, 2020.

[68] H. Zhou, J. Lan, R. Liu, and J. Yosinski, "Deconstructing lottery tickets: Zeros, signs, and the supermask," in *Advances in Neural Information Processing Systems*, 2019, pp. 3597–3607.

[69] Y. LeCun, C. Cortes, and C. Burges, “Mnist handwritten digit database,” *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, vol. 2, 2010.

[70] A. Gaier and D. Ha, “Weight agnostic neural networks,” in *Advances in Neural Information Processing Systems*, 2019, pp. 5364–5378.

[71] F. Chollet *et al.*, *Keras*, `https://keras.io`, 2015.

[72] M. Abadi, A. Agarwal, P. Barham, *et al.*, *Tensorflow: Large-scale machine learning on heterogeneous systems*, Software available from tensorflow.org, 2015. [Online]. Available: `http://tensorflow.org/`.

[73] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[74] V. Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” in *ICML*, 2010, pp. 807–814.

[75] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: `http://arxiv.org/abs/1412.6980`.

[76] Y. N. Dauphin and Y. Bengio, “Big neural networks waste capacity,” *arXiv preprint arXiv:1301.3583*, 2013.

[77] Y. LeCun, J. S. Denker, and S. A. Solla, “Optimal brain damage,” in *Advances in neural information processing systems*, 1990, pp. 598–605.

[78] J. Frankle and M. Carbin, “The lottery ticket hypothesis: Finding sparse, trainable neural networks,” *arXiv preprint arXiv:1803.03635*, 2018.

[79] A. Krizhevsky, G. Hinton, *et al.*, “Learning multiple layers of features from tiny images,” *Master’s thesis, Department of Computer Science, University of Toronto*, 2009.

[80] P. Chrabaszcz, I. Loshchilov, and F. Hutter, “A downsampled variant of imagenet as an alternative to the cifar datasets,” *arXiv preprint arXiv:1707.08819*, 2017.

[81] S. Falkner, A. Klein, and F. Hutter, “Bohb: Robust and efficient hyperparameter optimization at scale,” *arXiv preprint arXiv:1807.01774*, 2018.

[82] L. Li and A. Talwalkar, “Random search and reproducibility for neural architecture search,” in *Uncertainty in artificial intelligence*, PMLR, 2020, pp. 367–377.

[83] H. Liu, K. Simonyan, and Y. Yang, “Darts: Differentiable architecture search,” *arXiv preprint arXiv:1806.09055*, 2018.

[84] X. Dong and Y. Yang, “Searching for a robust neural architecture in four gpu hours,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1761–1770.

[85] ——, “One-shot neural architecture search via self-evaluated template network,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 3681–3690.

[86] G. Landrum, *Rdkit: A software suite for cheminformatics, computational chemistry, and predictive modeling*, 2013.

[87] F. Pedregosa, G. Varoquaux, A. Gramfort, *et al.*, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[88] S. Otsuka, I. Kuwajima, H. Junko, Y. Xu, and M. Yamazaki, “Polyinfo: Polymer database for polymeric materials design,” in *2011 International Conference on Emerging Intelligent Data and Web Technologies*, 2011, pp. 22–29. DOI: 10.1109/EIDWT.2011.13.

[89] J González-Benito, E Castillo, and J. Cruz-Caldito, “Determination of the linear coefficient of thermal expansion in polymer films at the nanoscale: Influence of the composition of eva copolymers and the molecular weight of pmma,” *Physical Chemistry Chemical Physics*, vol. 17, no. 28, pp. 18 495–18 500, 2015.

[90] R. Simha and R. Boyer, “On a general relation involving the glass temperature and coefficients of expansion of polymers,” *The Journal of Chemical Physics*, vol. 37, no. 5, pp. 1003–1007, 1962.

[91] A. Bondi and J. W. Sons, *Physical Properties of Molecular Crystals, Liquids, and Glasses*, ser. Wiley series on the science and technology of materials. Wiley, 1968, ISBN: 9780471087663.

List of publications

[P1] G. Lambard and E. Gracheva, “SMILES-X: Autonomous molecular compounds characterization for small datasets without descriptors,” *Machine Learning: Science and Technology*, vol. 1, no. 2, p. 025 004, 2020. DOI: 10.1088/2632-2153/ab57f3.

[P2] E. Gracheva, “Trainless model performance estimation based on random weights initialisations for neural architecture search,” *Array*, vol. 12, p. 100 082, 2021, ISSN: 2590-0056. DOI: https://doi.org/10.1016/j.array.2021.100082.

[P3] E. Gracheva, G. Lambard, S. Samitsu, K. Sodeyama, and A. Nakata, “Prediction of the coefficient of linear thermal expansion for the amorphous homopolymers based on chemical structure using machine learning,” *Science and Technology of Advanced Materials: Methods*, vol. 1, no. 1, pp. 213–224, 2021. DOI: 10.1080/27660400.2021.1993729.