Downloaded from orbit.dtu.dk on: Apr 18, 2024

DTU Library

Deep Learning Methods for Screening of Molecules and Materials

Jørgensen, Peter Bjørn

Publication date:
2019

Document Version
Publisher's PDF, also known as Version of record

Link back to DTU Orbit

Citation (APA):
Jørgensen, P. B. (2019). *Deep Learning Methods for Screening of Molecules and Materials*. Technical University of Denmark.

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Ph.D. Thesis
Doctor of Philosophy

DTU Compute
Department of Applied Mathematics and Computer Science

Deep Learning Methods for Screening of Molecules and Materials

Peter Bjørn Jørgensen

Kongens Lyngby 2019

DTU Compute
Department of Applied Mathematics and Computer Science
Technical University of Denmark

Matematiktorvet
Building 303B
2800 Kongens Lyngby, Denmark
Phone +45 4525 3031
compute@compute.dtu.dk
www.compute.dtu.dk

Summary (English)

Screening of molecules and materials is important in a number of scientific fields, e.g. discovery of new drugs and better materials for solar cells and batteries. Within the last few years machine learning and deep learning methods in particular have shown remarkable results in a wide range of scientific fields. In this thesis we develop new deep learning methods for screening of molecules and materials.

How to appropriately represent molecules and materials for machine learning methods is an ongoing field of research. Molecular graphs for molecules and quotient graphs for crystal structures are general representations but they are difficult to handle for standard machine learning algorithms due to the non-vectorial nature of these representations. In the first part of the thesis we develop new methods based on graph neural networks, which is a class of deep learning models that can be formulated as message passing on graphs. We propose to extend existing models with an edge update network that improves upon previous state of art results on the three publicly available datasets. In many screening applications we do not know the exact positions of the atoms of a new candidate molecule or material. We therefore propose to use a graph neural network model without access to the exact interatomic distances and show that it is still possible to accurately predict the formation energy of materials using only the connectivity of the atoms. In some cases we know both the connectivity and the symmetries of a material and we propose a new method for encoding the local symmetry of a material into its graph representation and show the efficacy of the method on publicly available materials datasets.

An important part of the screening process is to suggest new candidate materials and molecules. In the second part of the thesis we review current state of the art in deep learning methods for generating molecules. We focus on a specific latent variable model called the variational autoencoder, which is a model that learns a vector space representation of a given dataset. We use the model to accelerate the screening process of molecules for polymer solar cells by designing a grammar representation of the molecules and optimise for the optical properties of interest in the learned latent space.

Finally we reflect on the results of this research and discuss directions for future research on deep learning for molecules and meterials.

Summary (Danish)

Screening af molekyler og materialer er vigtigt for en række videnskabelige områder, for eksempel opdagelsen af nye former for medicin, og bedre materialer for solceller og batterier. Indenfor de sidste få år har maskinlæring og især dybe neurale natværk vist lovende resultater indenfor flere videnskabelig felter. I denne afhandling udvikler vi nye metoder, baseret på dybe neural netværk, til screening af molekyler og materialer.

Hvordan man skal repræsentere molekyler og materialer for anvendelse i maskinlæringsalgoritmer er stadigt et aktivt forskningsområde. Molekylære grafer og kvotientgrafer for krystalstrukturer er generelle repræsentationer, men de er svære at behandle for standard maskinlæringsalgoritmer på grund af deres ikke-vektorielle form. I den første del af afhandlingen udvikler vi nye metoder baseret på neurale grafnetværk, som er en klasse af dybe neurale netværksmodeller der kan formuleres some beskedudveksling på grafer. We foreslår at udvide eksisterende modeller med et kantopdateringsnetværk hvilket forbedrer de hidtil bedste resultater på tre offentligt tilgængelige datasæt. I mange screeningsapplikationer kender vi ikke atomernes nøjagtige positioner for nye molekyler eller materialer. Derfor foreslår vi at bruge neurale grafnetværk uden adgang til de nøjagtige interatomare afstande og viser at det stadigt er muligt at forudsige dannelsesenthalpien nøjagtigt udelukkende baseret på atomernes bindinger. I nogle tilfælde kender vi både bindingerne og symmetrierne af et materiale og vi foreslår en ny metode til at indkode de lokale symmetrier af et materiale i dets grafrepræsentation og vi viser metodens effektivitet på offentligt tilgængelige materiale-datasæt.

En vigtig del af screeningsprocessen er at finde på nye kandidater af materialer og molekyler. I anden del af afhandlingen giver vi et overblik over de nyeste metoder til generering af molekyler. Vi fokuserer på en specifik latent variabel model kaldet variationel autoenkoder, som er en model der lærer en latent vektorrumsrepræsentation af et givet datasæt. Vi bruger modellen til at accelerere screeningsprocessen af molekyler til polymer-solceller ved at designe en grammatisk repræsentation af molekylerne og optimere de optiske egenskaber i det lærte latente rum.

Til slut reflekterer vi over resultaterne og diskuterer mulige retninger for fremtidig forskning indenfor maskinlæring på molekyler og materialer.

Preface

This PhD thesis was prepared at the department of Applied Mathematics and Computer Science at the Technical University of Denmark in partial fulfillment of the requirements for acquiring a PhD degree at Technical University of Denmark. The PhD project was funded by a DTU Compute PhD Scholarship and the work carried out at the ITMAN graduate school at DTU Compute.

The main results of the work is disseminated via four papers, of which three are published and one is ready for submission. The papers are an integrated part of the thesis and are included at the end. Additionally two co-authored papers are not included in the thesis. The introductory chapters of the thesis will introduce the topics that are addressed in the papers and summarise the most important results.

Kongens Lyngby, March 14, 2019

Peter Bjørn Jørgensen

Acknowledgements

I am very grateful for having received the DTU Compute scholarship which has allowed me to dig into machine learning research with a very high degree of scientific freedom and room for failures (and a few fruitful ventures). I also acknowledge the donation of GPUs from NVIDIA.

Thanks to all the people at the Cognitive Systems sections including my supervisors Mikkel N. Schmidt and Morten Mørup for the guidance and my office mates Rasmus Bonnevie, Michael Riis Andersen, Rasmus Larsen and Lærke Karen Krohne for the company, scientific discussions and for bearing with my wet cycling clothes. I would also like to thank the people at the Machine Learning group at Technical University of Berlin for being welcoming and letting me spend for four months with the group during the autumn in 2019, and thanks to my collaborators at DTU Physics Karsten W. Jacobsen, Estefanía Garijo del Río, Kristian Sommer Thygesen, Suranjan Shil and at DTU Energy Murat Mesta and Juan Maria García Lastra.

Special thanks to my father who is no longer here, but has always been a great inspiration and to my mother for the love and support throughout the years. Thanks to my nerdy brothers Jeppe and Asger for all the discussions, support, fights and fun.

Contents

Summary (English) i

Summary (Danish) iii

Preface v

Acknowledgements vii

Contents ix

1 Introduction 1
- 1.1 Machine Learning Opportunities in Molecular Science 1
- 1.2 Outline 2

2 Representation of Molecules and Materials 5
- 2.1 Molecular Descriptors 5
- 2.2 Materials Descriptors 7

3 Deep Learning on Graphs 13
- 3.1 Neural Message Passing on Graphs 14
- 3.2 Encoding Symmetry Information Locally 20
- 3.3 Contributions 21

4 Generative Models 25
- 4.1 Latent Variable Models 26
- 4.2 Variational Autoencoder 27
- 4.3 Generating new Molecules 31
- 4.4 Contributions 36

5 Conclusion 37
- 5.1 Open Questions and Future Directions 38

Bibliography 41

A Neural Message Passing with Edge Update Networks for Predicting Properties of Molecules and Materials **51**

B Materials property prediction without atomic positions using graph neural networks **63**

C Deep Generative Models for Molecular Science **77**

D Machine learning-based screening of complex molecules for polymer solar cells **87**

CHAPTER 1

Introduction

1.1 Machine Learning Opportunities in Molecular Science

In recent years machine learning and especially deep learning methods have shown impressive results in image processing, natural language processing and playing games. In the fields of physics and chemistry large amounts of data are being generated which is a perfect fit for data hungry machine learning algorithms. This match is reflected in the increase in the number of publications using machine learning for chemistry and materials science as shown in Figure 1.1. There are several different problems

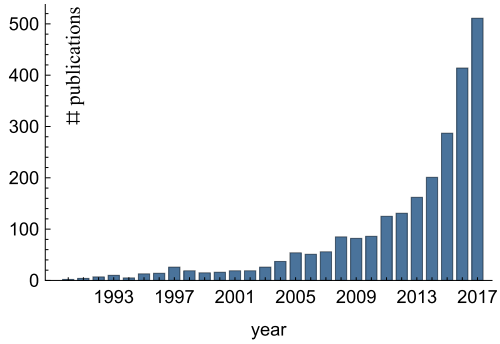


Figure 1.1: Number of publications per year from a web of science search for articles with topics from machine learning and either chemistry or materials. Figure taken from [RLB18]

.

that are being approached with machine learning methods. The following is a short description of a few notable examples.

Potential energy surface for molecular dynamics. Molecular dynamics simulations are used for studying the movement of molecules and atoms over short time periods. The potential energy of the system is calculated for different configurations as the system evolves and machine learning can be used to inter-

polate between these calculated configurations. Often the models are trained on a single trajectory. [Beh11; Beh16]

Learning exchange correlation functional. The accuracy of density functional theory (DFT) calculations depends on the chosen exchange correlation functional, which models how the particles of the system interact. The exact exchange correlation functional exists but is not known and a vast amount of research goes into finding a good approximation for the functional. Machine learning could be used to learn the functionals from data. See [Liu+17; Nag+18; RST18; Bro+17] for examples in this direction.

Predicting the outcome of chemical reactions. A chemical reaction is a transformation of a set of chemical substances (reactants) to another set of chemical substances (products) and is for example used for determining how to synthesize a target molecule. A chemist may predict the products of a reaction based on chemical rules and heuristics. Machine learning can be used to automatically discover these and other rules from large databases of chemical reactions. See for example [KB11; Sch+17b; Foo+18].

Chemical compound space to property prediction. In this setting we are concerned with learning a mapping from a molecule or material to a property of interest. The available input information could be just the composition (number of atoms and their atomic numbers) [Mer+14], the chemical graph (atoms and bond types) [Han+15; Gil+17; Col+18], or the full spatial information including the positions of the atoms [Fab+17; Col+18]. We introduce deep learning methods based on the chemical graph in chapter 3.

Generate molecules with desired properties. Throughout the literature this is also known as inverse design because the usual approach is to start out with a pool of candidate molecules and then select or optimise towards the desired property. We cover this topic in chapter 4.

In this thesis we are concerned with virtual screening of molecules and materials and focus on the last two problems above. Virtual screening is the process of evaluating a large set of candidate molecules or materials in order to select a much smaller set to be synthesized and validated by experiment. Often virtual screening is performed in several filtering steps in which candidates are rejected in each step often referred to as a screening funnel as illustrated in Figure 1.2. Machine learning methods can play a key role in one or several of these steps and DFT calculations are often used within the last steps of the process.

1.2 Outline

The rest of the thesis is arranged into three introductory chapters. In chapter 2 we introduce different ways of representing molecules and materials for use in machine

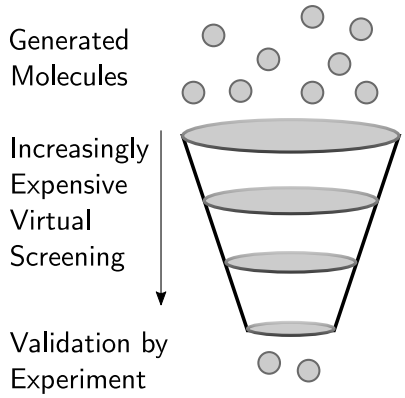


Figure 1.2: Screening of molecules and materials is often portrayed as a funnel with several filtering steps in which candidates are rejected in each step.

learning. The hand-crafted representations are often known as descriptors or fingerprints and the development of new descriptors have driven the field forward for many years. When we talk about materials in this thesis we are concerned with arrangements of atoms in repeating spatial structures. These infinite structures often require some extra thoughts in the design of descriptors and we therefore address materials descriptors separately in Section 2.2.

One of the core parts of this thesis is machine learning models on molecular graphs and this topic is introduced in chapter 3 and the main results are documented in the papers *Neural Message Passing for Edge Update Networks* and *Materials property prediction without atomic positions using graph neural networks* in Appendix A and B respectively. The other main part of the thesis is concerned with generative models and how to use them to generate new molecules with desired properties and this topic is introduced in Chapter 4. Again the main results are documented in two papers *Deep Generative Models for Molecular Science* and *Machine learning-based screening of complex molecules for polymer solar cells* which are included in Appendix C and D respectively. Finally we sum up the work and give pointers for future research in Chapter 5.

CHAPTER 2

Representation of Molecules and Materials

This chapter is a mini review of different representations for molecules and materials for machine learning applications.

2.1 Molecular Descriptors

A molecule is fully defined by the atom types and their positions and therefore all properties of interest can be derived from only this information. Traditionally machine learning for molecules has revolved around developing hand crafted vector representations and use them in methods such as kernel ridge regression, Gaussian process regression, support vector machines and feed-forward neural networks.

In order to generalise well it is desired that the descriptors are invariant to rotation and translation of the molecules. Depending on the task it is also important that the descriptors are able to distinguish between molecules with different target values. A widely used descriptor is the Coulomb matrix [Rup+12; Mon+12] with entries:

$$C_{ij} = \begin{cases} 0.5 Z_i^{2.4} & \text{for } i = j \\ \frac{Z_i Z_j}{||\mathbf{r}_i - \mathbf{r}_j||} & \text{for } i \neq j \end{cases} \tag{2.1}$$

Where Z_i is the charge (atomic number) of atom i and $\mathbf{r}_i$ is it's position. Thus, invariance to translation and rotation is achieved by using the distance between atoms rather than absolute positions. Given the coulomb matrix and by fixing the position of one atom it is possible to solve a system of equations to obtain the original atom numbers and positions (up to rotation and translation) and the descriptor thus has no loss of information. One of the disadvantages of this representation is that it is not invariant to permutation of the atom indices. Solutions proposed by Montavon et al. [Mon+12] and Rupp et al. [Rup+12] is to sort the rows of the Coulomb matrix based on the norm of the rows or to use the sorted eigenvalues of the matrix.

Using just the eigenvalues is a dimensionality reduction which may introduce noise. Sorting the rows causes the descriptor to be discontinuous to small changes, because a small change can swap two rows that are very different but have similar vector norm. In the bag-of-bonds [Han+15] representation we keep only the entries of the coulomb matrix that corresponds to bonds between a pair of atoms and the entries are arranged into different "bags" depending on the atom types. Again the entries are sorted within each bag. The authors define a distance function between two of these vector representations which means that two molecules are deemed similar if they have a similar set of bond distances. In the bonds and angles representation by Huang and Lilienfeld [HL16] this idea is further developed by also including angles between triplets of atoms and torsions between quadruplets, which gives a more detailed notion of similarity. A different representation of a molecule is a histogram of the angles and distances (HAD) [Fab+17]. Again invariance to translation and rotation is achieved by being dependent on the distance between atoms and angles between triplets of atoms rather than absolute positions and angles and because the distances and angles are projected onto histograms for the different atom types there is no ordering problem.

In many applications the exact position of atoms is not available and the molecules are instead specified as a molecular graphs, i.e. a graph where each vertex is annotated with an atom type (C, H, O etc.) and each edge is annotated with a bond type (single, double or triple bond). Examples of descriptors derived from molecular graph features are Extended-Connectivity fingerprints (ECFP) [RH10] and the shape indices proposed by Randić [Ran01]. The ECFP descriptor is constructed by counting the presence of certain substructures in the graph and the shape indices are constructed by counting the number of different paths and walks on the graph.

All the descriptors above are global in the sense that they describe the molecules as a fixed size vector regardless of the size of the molecules. For the Coulomb matrix and bag-of-bonds this is achieved through zero-padding, which means that the maximum size of the molecules needs to be known beforehand.

Other descriptors focus on characterising the *chemical environment* from the perspective of a single atom. A prime of example of this is the atom-centered symmetry functions (ACSF) [Beh11] where a number of predefined functions capture radial and angular information about an atom's neighborhood and a neural network is then used to assign an energy contribution to each atom. Another example of characterising local chemical environment is the Smooth Overlap of Atomic Positions (SOAP) descriptor [BKC13], in which a Gaussian distribution function is placed on each neighboring atom. For two atoms the similarity between the distribution of neighbours is used to construct a chemical environment kernel. A matching procedure can be built on top to define the similarity between two molecules [De+16]. Recently Huang and Lilienfeld [HL17] proposed a local descriptor called Amons based on the molecular graph. Here the graph is broken into commonly used building blocks and an energy contribution is assigned to each substructure through kernel ridge regression.

It is also possible for a machine learning method to work directly on the set of atom types and positions or on the molecular graph. These are not vectorial data and

the methods must therefore be engineered for this data type. A widely used approach is to encode the molecular graph into a string representation such as SMILES [Wei88] or InChI [Hel+15]. The biggest benefit of this from a machine learning perspective is that we can employ the toolbox of methods from natural language processing. However, SMILES and InChI are not natural languages (for example there is no or very little redundancy) and this is something to be aware of when employing methods from natural language processing.

We summarise the molecular descriptors in Table 2.1. Note that the descriptors are just a few examples to highlight different approaches taken in the field and not an exhaustive list.

2.2 Materials Descriptors

In the previous section we reviewed different representations for molecules and in this section we will do the same for materials. We limit our scope to crystals. Formally, a crystal structure is a time-invariant three-dimensional arrangement of atoms or molecules on a lattice [DM07]. A space lattice is the geometrical image of the operation of the translation operators on the node at the origin [DM07].

We can represent a crystal structure with a set of atom types and positions and a set of three lattice vectors $\mathbf{a}, \mathbf{b}, \mathbf{c}$ in $\mathbb{R}^3$. The vectors do not need to be orthogonal or of equal length. The vectors points to the corner of the unit cell "box", i.e. we

	Type				
Information	Loc.	Glob.	NV	Descriptor	Reference
			x	Atom types and positions	
		x		Coulomb matrix	[Rup+12]
Spatial		x		Bag-of-Bonds	[Han+15]
Structure		x		Bonds and Angles (BAML)	[HL16]
		x		Histogram (HAD)	[Fab+17]
	x			SOAP	[BKC13; De+16]
	x			ACSF	[Beh11]
			x	Molecular graphs	
Connectivity			x	SMILES	[Wei88]
Only		x		ECFP fingerprints	[RH10]
	x			Amons	[HL17]

Table 2.1: Classification of molecular descriptors into the classes local (Loc.), global (Glob.) and non-vectorial (NV). We can also divide them into the two categories whether they use the three dimensional spatial structure of the atoms or they use the connectivity only.

can get to all corner points by integer linear combinations of the three basis vectors $\mathbf{t} = u\mathbf{a} + v\mathbf{b} + w\mathbf{c}$ as shown for two dimensions in Figure 2.1.

For materials we are also interested in designing descriptors that are invariant to translation and rotation. Furthermore, because the choice of unit cell is not unique the descriptors should also be invariant to the choice of unit cell. The SOAP descriptor introduced above is a local descriptor and by introducing a cutoff distance it can also be used for crystal structures [BKC13]. The Coulomb matrix is not directly applicable for materials but Faber et al. [Fab+15] have proposed three different extensions that takes the periodicity into account, Ewald sum matrix, Extended Coulumb Matrix and Sine Matrix. Another representation based on interatomic distances is the partial radial distribution function (PRDF) representation [Sch+14]. It considers the distribution of pairwise distances between two atom types. We can consider this representation as a collection of smeared histograms for the pairwise distances between all atom type pairs.

Crystalline materials can be classified by their symmetry and every material belongs to one of 230 space groups [San93]. The symmetry puts some restrictions on where the atoms can be, which is characterised by the socalled Wyckoff sites [San93]. This was recently utilised by Jain and Bligaard [JB18] to build a model in which the atomic species and Wyckoff sites are encoded into an indicator matrix and an attention-based neural network is trained on this representation. This mitigates the need to know the exact spatial structure, because only the spacegroup and the atomic species and their Wyckoff sites are needed.

Graph representations are more challenging for crystal structures than for molecules because the bonds between atoms are less well defined. One way to obtain bonds from atomic positions is to use a constant cutoff distance such that a pair of atoms is connected if they are within a certain distance of each other. This opens up the question of how to choose the constant cutoff distance. We could also base the cutoff distance on the covalent radii of the atoms in a pair. Another popular choice that mitigates the choice of cutoff distance is to use Voronoi tessellation. A Voronoi diagram is a

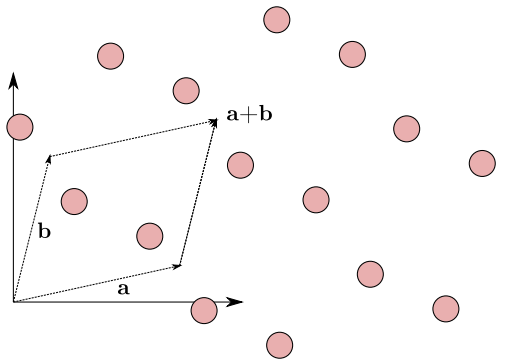


Figure 2.1: Example unit cell with basis vectors **a** and **b**.

partitioning of the space into cells. In each cell the distance to the point (atom) that generated the cell is shorter than to any other point (atom). An example in two dimensions is shown in Figure 2.2. We can now say that two atoms are bonded if they share a face in the Voronoi diagram. An interesting property of this method is that it is invariant under rotation, translation and scaling of the atom coordinates. We can further build upon this method. For example we could remove bonds between two atoms if they are too far from each other or we can remove bonds if a straight line between two atoms does not intersect the face shared between the two atoms as done by Malins et al. [Mal+13]. An inherent problem with the Voronoi graph construction method is that small perturbations of the atom positions can lead to different graphs. This instability has even been exploited to classify local structures by classifying different types of instability [LHS15]. We can alleviate the problem by removing bonds if the size of a shared face is below a given threshold, but the threshold is now a parameter that must be chosen. Ward et al. [War+17] have proposed a descriptor for crystal structures based on the Voronoi diagram. They construct the Voronoi diagram, extract different features and summarise different statistics of the features in a vector. All the features are invariant to scaling of the material, e.g. the distance between neighbouring atoms are normalised by the average distance between all neighbours in the material. In their work they train a random forest algorithm on top of this descriptor to predict the heat of formation of new materials. The materials descriptors are summarised in table 2.2.

2.2.1 Quotient Graphs

In the discussion of graph representations above we have neglected the fact that the graphs are infinite because of the periodicity of the structures. In the graph literature the actual physical crystalline structures are more accurately referred to as

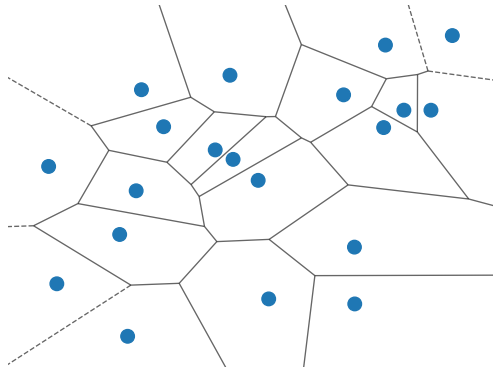


Figure 2.2: Example of a two dimensional Voronoi tessellation

Information	Type Loc.	Glob.	NV	Descriptor	Reference
			x	Atom positions and unit cell	
Spatial		x		PRDF	[Sch+14]
Structure		x		Extended Coulomb matrix	[Fab+15]
		x		Voronoi Diagram Features	[War+17]
	x			SOAP	[BKC13; De+16]
Connectivity			x	Quotient Graphs	[CHK84]
Symmetries	x			Wyckoff-species matrix	[JB18]

Table 2.2: Classification of materials descriptors into the classes local (Loc.), global (Glob.) and non-vectorial (NV). We can also divide them into the two categories whether they use the three dimensional spatial structure of the atoms or they use only the connectivity

three-periodic *networks* (even though all real-world crystals are finite). Three-periodic networks (or nets) are connected graphs which permit embeddings having a threefold periodicity [CHK84]. A network is a physical object with nodes and links between nodes and a graph is a mathematical abstraction that not necessarily has a spatial embedding. A particular finite graph representation for periodic nets was introduced by Chung, Hahn, and Klee [CHK84]. The mapping between periodic nets and these so called quotient graphs has been studied by several authors [CHK84; Eon98; Kle04]. In this section we will highlight some of its properties and challenges.

The quotient graph can be constructed from the smallest possible unit cell by mapping each atom in the unit cell to a vertex of the finite graph. Thus each atom is a node lattice $\mathbf{P}_i$ that is mapped to one vertex P_i of the finite graph. Similarly each link lattice that is a link between $\mathbf{P}_i$ and $\mathbf{P}_j$ is mapped to an edge between P_i and P_j. Note that the finite graph can have loops and multiple edges between a pair vertices. Two periodic nets that maps to the same (isomorphic) finite graph using this procedure is shown in Figure 2.3. This mapping preserves some properties such as the number of neighbours for each atom and their types, but we lose track of which particular atoms are connected [Kle04]. To preserve this information we need to reintroduce the coordinate system described by a set of basis vectors $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$. We now choose an arbitrary point in each lattice as origin and label the point $P_i(00)$. Using the basis vectors we can describe any point in the lattice with the origin plus an integral linear combination of the basis vectors, which is written as $P_i(01)$ meaning the point $P_i(00) + \mathbf{a} \cdot 0 + \mathbf{b} \cdot 1$. The edges of the quotient graph are then labeled such that a link from $\mathbf{P}_i(uvw)$ to $\mathbf{P}_j(rst)$ is labeled with the triplet $r-u, s-v, t-w$ and the edge is directional from P_i to P_j. The convention is to leave out the annotation for edges with (000) and there is also no need to use directional edges for this type of edge, because the direction of the edge does not change the label. This labeling is shown

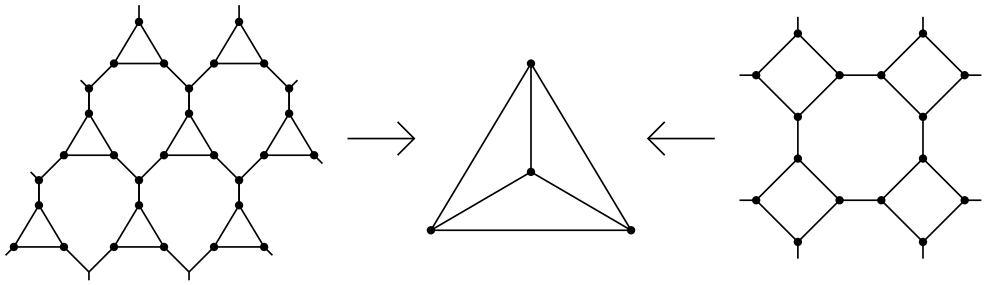


Figure 2.3: Two different two-periodic nets map to the same (isomorphic) unlabeled quotient graph. Based on figure 2 in [Kle04].

for the two dimensional examples in Figure 2.4. Importantly a three-periodic net is determined up to isomorphism by the quotient graph and the embedding of the three-periodic net is uniquely determined by the quotient graph and the three basis vectors [CHK84]. However, the labeling of a quotient graph is not uniquely determined by the three-periodic net because many labellings are equivalent, i.e. they determine the same three-periodic net. The following operations create equivalent quotient graphs:

1. Reversal of an edge and inversion of its label, i.e. an edge from P_i to P_j with label $(1, 1, 1)$ is equivalent to an edge from P_j to P_i with label $(-1, -1, -1)$. This type of equivalence can be avoided if only non-negative labels are allowed.

2. Change of coordinate system. The choice of origin and basis vectors do not change the embedding, but it may change the labeling of the quotient graph.

3. Applying an automorphism of the quotient graph.

This must be taken into account when designing algorithms working on quotient graphs. Quotient graphs have been used for designing unique identifiers of crystal structures [DO03; Del+17]. The quotient graph is translated into a text representation and uniqueness is solved by choosing a canonical ordering of the translation. The link between crystal topology, which can be represented as quotient graph without basis vectors, and structural symmetries has been examined by Thimm [Thi09] and it shows how the relative position of nodes is constrained by certain symmetries. A similar study has been carried out by Eon [Eon11]. These work are beyond the scope of this thesis but they might be important pillars for generative modeling of crystal structures. For example the model could model connectivity and symmetries using a quotient graph and the atomic positions could be more or less derived from only this information. We also utilise unlabeled quotient graphs when we do neural message passing in Chapter 3.

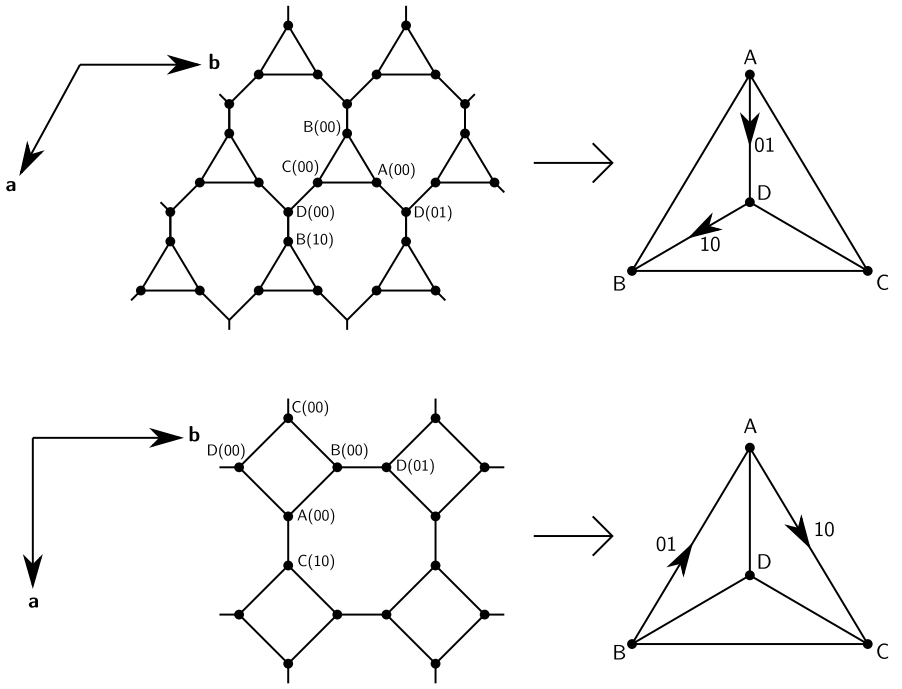


Figure 2.4: The two-periodic nets map to different (non-isomorphic) labeled quotient graph. Based on figure 3 in [Kle04].

CHAPTER 3

Deep Learning on Graphs

In the previous chapter we introduced the different representations of molecules and materials. The molecular graph is a general representation, but not well suited for standard machine learning algorithms that are designed for vectorial inputs. In this chapter we explore a family of machine learning algorithms that operates on graphs. One of the key ingredients is the use of artificial neural networks. Artificial neural networks is a family of mathematical models inspired by the biological brains and their invention is credited to Rosenblatt [Ros62] with his work on the multilayer perceptron. Efficient training of these models through back-propagation was promoted in the 1980's by different research groups [Lec85; RHW86]. The term *deep learning* refers to the practice of using neural networks with many layers. The current deep learning popularity was ignited by AlexNet [KSH12], which won the 2012 ImageNet image classification competition by a large margin using a combination of convolutional neural network layers, dropout regularisation and training on graphics processing units (GPUs) with an augmented data set. Since then deep learning methods have become popular for a wide range of applications.

The virtue of deep learning is the ability to learn from data with no (or very little) feature engineering. The universal approximation theorem states that a neural network with one hidden layer can approximate any function (under mild conditions) if the number of hidden nodes is sufficiently large [Cyb89; HSW89; Hor91]. More recently it has also been shown that residual networks with just one hidden node per layer is also a universal approximator if the number of layers is sufficiently large [LJ18]. This high abstraction level view of neural networks as universal approximators can be useful when designing machine learning models based on neural networks. Neural message passing on graphs is one example of this approach. However, whether a neural network model can solve a given problem and generalise well to new examples depends on the choice of model architecture, optimisation tricks and computational resources.

3.1 Neural Message Passing on Graphs

A graph represents a set of objects in which some of the objects have a relation in some sense. The objects are represented as vertices and the relations as edges between those vertices. The idea behind graph neural networks is to exploit the relational structure of the data and use it as an inductive bias for a learning algorithm. Inductive biases are factors that lead a learner to favor one solution over another when the two solutions fit the data equally well [Gri+10]. In Bayesian modeling the prior distribution is an example of an inductive bias. In neural network models trained with stochastic gradient descent the inductive biases arise from the choice of model architecture, weight initialisations and the cost function, which for example can include regularisation terms to favour smoother solutions. When using graph neural networks to learn a mapping from a graph to a target variable we favour solutions that can be explained by the relation (or interaction) between the objects of the graph. As part of the learning algorithm the model should learn how these objects interact.

The first graph neural networks were invented by Gori, Monfardini, and Scarselli [GMS05]. To describe the different architectures we will use the message passing framework introduced by Gilmer et al. [Gil+17], who also showed how a number of graph neural networks can be cast in this framework. The framework can of course be generalised further as shown by Battaglia et al. [Bat+18], but we will stick to the slightly simpler framework, because it is expressive enough to describe the algorithms we deal with here.

3.1.1 Message Passing Equations

We now introduce the message passing framework devised by Gilmer et al. [Gil+17] and also described in our paper *Neural Message Passing with Edge Update Networks for Predicting Properties of Molecules and Materials* in Appendix A. Denote the graph G with vertex features x_v and edge features ε_{vw} for an edge from vertex v to w. Each vertex has a hidden state h_v^t at "time" step t and each edge has a hidden state e_{vw}^t. They are updated in a number of interaction time steps T. The hidden states of the vertices are updated by first receiving messages from neighbouring vertices, which are computed by the message function $M_t(\cdot)$. The vertex state is updated by a state transition function $S_t(\cdot)$.

$$m_v^{t+1} = \sum_{w \in N(v)} M_t(h_v^t, h_w^t, e_{vw}^{t+1}), \tag{3.1}$$

$$h_v^{t+1} = S_t\left(h_v^t, m_v^{t+1}\right), \tag{3.2}$$

where $N(v)$ denotes the neighbourhood of v, i.e. the vertices that have an edge to v. In some models the edges are updated by an edge update function $E_t(\cdot)$ that can depend on the previous edge state and the sending and receiving vertices:

$$e_{vw}^{t+1} = E_t\left(h_v^t, h_w^t, e_{vw}^t\right). \tag{3.3}$$

After T interaction steps the hidden state of the nodes is contained in h_v^T and the state of the edges in e_{vw}^T. A readout function $R(\cdot)$ is then applied which maps the set of vertex states and edge states to a single entity.

$$\hat{y} = R\left(\left\{h_v^T \in G, e_{vw} \in G\right\}\right). \tag{3.4}$$

The readout function operates on the set and must thus be invariant to permutations of the final vertex states and the final edge states. The message function M_t, state transition function S_t, edge update function E_t and the readout function R are all implemented as neural networks with trainable parameters. In some models, for example Deep Tensor Neural Network [Sch+17a] and the graph neural network [Sca+09] the weights are shared across the interaction steps, i.e. $M_t \equiv M$, $S_t \equiv S$ and $E_t \equiv E$ for all time steps t. Another interesting feature of the graph network proposed by Scarselli et al. [Sca+09] is that the transition from h^t to h^{t+1}, where h^t is a concatenation of all state vectors $\{h_v^t\}$, is designed to be a contraction map, which means that the interaction steps can be run until convergence rather than for a fixed number of time steps.

Often the readout function is a sum over the vertices only. Another popular readout function is the so called set2set readout function proposed by Vinyals, Bengio, and Kudlur [VBK15]. It is a recurrent neural network, specifically a long short-term memory network (LSTM) [HS97], that evolves a global hidden state q^τ using attention:

$$q^\tau = \mathrm{LSTM}(q^{\tau-1}, r^{\tau-1}), \tag{3.5}$$

$$u_v^\tau = f(h_v^T, q^\tau), \tag{3.6}$$

$$a_v^\tau = \frac{\exp(u_v^\tau)}{\sum_w \exp(u_w^\tau)}, \tag{3.7}$$

$$r^\tau = \sum_v a_v^\tau h_v^T, \tag{3.8}$$

$$\hat{q}^\tau = \left[q^t \, r^\tau\right], \tag{3.9}$$

where $\mathrm{LSTM}(x, y)$ is a LSTM network with previous output x and input y. The function $f(\cdot, \cdot)$ is in this case a function that maps a pair of equal length vectors to a scalar (for example a dot product) and the final output state $\hat{q}^\tau$ is the concatenation of the LSTM hidden state and a weighted sum of the node states.

3.1.2 Neural Message Passing with Edge Updates

Several graph neural networks [Bru+13; Duv+15; Li+15; Bat+16; Kea+16; Sch+17a] can be described by the message passing framework described above as shown by Gilmer et al. [Gil+17]. In our paper in Appendix A we show how the successful SchNet model [Sch+18] also fits into this framework and we further extend the model by including an edge update network. We summarise the architecture in this section.

As vertex input features, z_v, we use the atomic numbers which are used to look up an embedding vector for each atomic number. The initial hidden state $h_v^0 = \ell(x_v)$ is the result of a lookup function $\ell : \mathbb{Z} \to \mathbb{R}^C$. The edge feature ε_{vw} is the distance between two atoms. In general the graph edges are directional but the distance from atom v to atom w is the same as the distance from w to v. The distances are turned into a vectorial edge state by expanding them in a series of exponentiated quadratic functions:

$$(e_{vw}^0)_k = \exp\left(-\frac{(\varepsilon_{vw} - (-\mu_{\min} + k\Delta))^2}{2\Delta^2}\right), k = 0 \dots k_{\max} \tag{3.10}$$

where $\mu_{\min}$, Δ, and $k_{\max}$ are chosen such that the centers of the functions covers the range of the input features. This can be seen as a soft 1-hot-encoding of the distances, which makes it easier for a neural network to learn a function where the input distance is uncorrelated with the output of the network if that is necessary. The edge state represents how two atoms interact. As an edge update network we use a two layer neural network and the input is a concatenation of the sending and receiving vertex states and the current edge state.

$$e_{vw}^{t+1} = E_t(h_v^t, h_w^t, e_{vw}^t) = g(W_{\text{E2}}^t g(W_{\text{E1}}^t(h_v^t; h_w^t; e_{vw}^t))). \tag{3.11}$$

This choice of edge update network means that the edge state for each of the two different directions between a pair of vertices becomes different after the first update. The flow of an edge update and vertex update is shown in Figure 3.1. The activation function $g(x)$ is the shifted soft-plus activation function $g(x) = \ln(\mathrm{e}^x + 1) - \ln(2)$. The message function is only a function of the sending vertex and can be written as

$$M_t(h_v^t, h_w^t, e_{vw}^t) = M_t(h_w^t, e_{vw}^t) = (W_1^t h_w^t) \odot g(W_3^t g(W_2^t e_{vw}^t)), \tag{3.12}$$

where $\odot$ denotes element-wise multiplication. The expression on the right-hand side of the element-wise multiplication can be seen as a filter generating function. This is analogous to the discrete filters used in convolutional neural networks for image processing. Because the filter depends on the distance encoded in the edge feature it becomes a radial filter rather than the directional filters used in image processing.

The state transition function is a two layer neural network on the sum of incoming messages and the result is added to the current hidden state as in Residual Networks [He+15]:

$$S_t\left(h_v^t, m_v^{t+1}\right) = h_v^t + W_5^t g(W_4^t m_v^{t+1}), \tag{3.13}$$

After a number of interaction steps T we apply a readout function for which we use a two layer neural network that maps the hidden representation to a scalar and finally we sum over the contribution from each atom, i.e.

$$R\left(\left\{h_v^T \in G\right\}\right) = \sum_{h_v^T \in G} W_7 g(W_6 h_v^T), \tag{3.14}$$

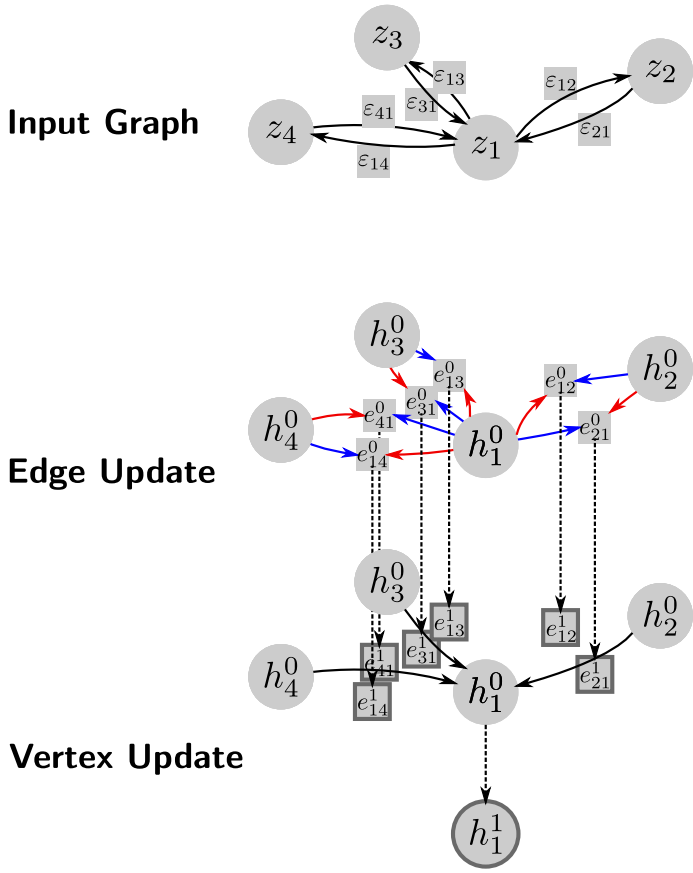


Figure 3.1: The hidden states of edges and vertices are updated in turns using message passing on the graph. In this illustration the hidden representation of the edges connected to h_1 are updated first and then h_1 is updated based on the new edge representation.

During training the cost function is the squared error between the output of the model and the target label. Because all the operations of the model are differentiable with respect to the model parameters the error can be back-propagated and the model is thus trained using gradient descent.

For molecular graphs other authors have found that using three interaction steps is sufficient [Sch+18; Gil+17] and for our applications we also did not see a gain in prediction accuracy when using more than three interaction steps. The only difference between our model presented here and the SchNet model is that the SchNet model uses the identity function instead of the two layer neural network for the edge update function in (3.11). As shown in our paper *Neural Message Passing with Edge Update Networks for Predicting Properties of Molecules and Materials* this minor change leads to a big improvement in prediction accuracy of formation energies and other molecular properties.

The model can also be used to predict forces because we can take the derivative of the output $\hat{y}$ with respect to the atom coordinates $r_v \in \mathbb{R}^3$ when rewriting the interatomic distances used as input in (3.10) as $\varepsilon_{vw} = ||r_v - r_w||$. Given target energy y and forces $F_1, \ldots, F_N$ the cost function is written as:

$$\ell = \rho(y - \hat{y})^2 + \frac{1-\rho}{N} \sum_{i=1}^{N} \left|\left| F_i - \left(-\frac{\partial \hat{y}}{\partial r_i} \right) \right|\right|^2, \tag{3.15}$$

where ρ is a weighting parameter that sets the tradeoff between energy errors and force errors. Modeling the forces is for example used in [Sch+18] for molecular dynamics simulations.

3.1.3 Limitations and Future Directions

The neural message passing models described above are invariant to translation and rotation of the molecules and materials. They are also invariant to relabeling of the vertices, i.e we can permute the indices of the vertices and get the same output. These invariances also come with caveats. In the following we describe two downsides of this model and the proposed solutions.

When we do message passing on a Voronoi graph without distance features we are effectively passing messages on the unlabeled quotient graph and as discussed in Section 2.2.1 this cannot describe the full connectivity of the graph, because we lose the identity of the vertices that the messages comes from. A simple example of two different graphs in which the vertex states will be the same even though their environments are different is shown in Figure 3.2. We have two vertex types A and B and in both cases B has two incoming edges from A and there is no way to tell whether the edges are coming from two different vertices (Figure 3.2 left) or the same vertex (Figure 3.2 right). This may seem like a contrived example, but another more realistic example arises when looking at crystals that consists of a single atom type. This is modeled as close packing of equal spheres. The most regular lattices that achieves the highest packing density are called face-centred cubic (fcc) and hexagonal

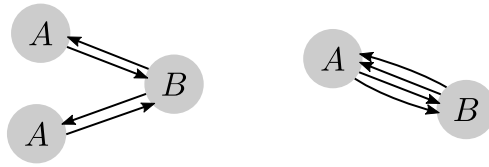


Figure 3.2: By using neural message passing on the two different graphs, the hidden states of A nodes and B nodes will be the same across the two different graphs

close-packed (hcp). These two packings can be constructed layer-wise as shown in Figure 3.3. We consider two spheres to be neighbours if they touch each other. In both cases every sphere has 12 neighbours and our message passing algorithm will not be able to differentiate between the two structures. This problem can be solved by covariant compositional networks proposed by Hy et al. [Hy+18] and Kondor et al. [Kon+18]. In this work the hidden state of the vertices keeps track of which vertices the messages originates from and the hidden representation thus co-varies with relabeling of the vertices. Even though the internal representation varies with the labeling, the final output of the network can be made invariant to the labeling. The model is demonstrated on the molecular graphs (no distance information) of Quantum Machines 9 (QM9) dataset [Rud+12; Ram+14] and it would be interesting to try to scale it up to bigger problems and datasets such as Open Quantum Materials Database (OQMD) [Saa+13; Kir+15] and Materials Project [Jai+13].

As discussed in Section 3.1.2 neural message passing with interatomic distances as edge features can also be interpreted as convolution with radial filters in the three dimensional space. In this setup we are limited to radial filters and scalar outputs and this puts some limitations on what can be modeled with this family of models. In the tensor field network proposed by Thomas et al. [Tho+18] the convolutional

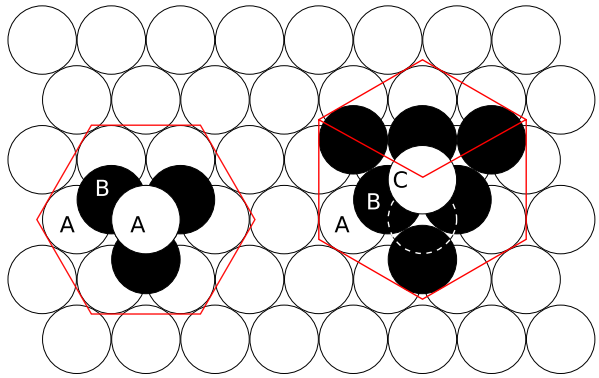


Figure 3.3: The hcp lattice (left) and fcc lattice (right). The letters indicate which layers are the same. (public domain figure from Wikipedia)

filters use spherical harmonics as basis functions together with a radial component defined by a neural network. This enables the vertex hidden states to be geometric tensors, e.g. one could attempt to learn the forces in a molecule as a three-dimensional vector for each atom. The tensors are equivariant to rotation, which means that they rotate in the same way as the molecule rotates. This is similar to how a convolutional neural network for images is equivariant to translations of the input image. The paper [Tho+18] outlines the theory behind it and show some examples on three dimensional shape classification and prediction of atom positions in molecules with a missing atom. It will be interesting to see how future applications will benefit from this framework.

3.2 Encoding Symmetry Information Locally

In virtual screening of materials the candidate materials are often generated by taking known stable materials and replacing some of the atoms with other atom types. In this application we do not know the exact spatial structure of the new material and we might approximate it by its connectivity and symmetries. Symmetry of materials is classified into 230 different space groups, which consists of the translational symmetry of the unit cell, the point group symmetry operations (rotation, reflection and rotoinversion) and screw axis and glide plane symmetry operations [DM07]. The space group number is thus a global identifier for the symmetry operations that are present, but the question is how to encode this information into the unlabeled quotient graph that we use as input for the graph neural networks discussed in this chapter. We could encode the information as a special "global node" feature in the graph and one-hot encode the space group number. In the paper B we propose a method for encoding the point group symmetry operations locally. The method is based on classifying the faces of the Voronoi diagram into the 10 different two dimensional point groups. For example a Voronoi cell for one of the atoms in Au_2CaSi_2 is shown in Figure 3.4. Each face of the cell corresponds to a bond to another atom and we would like to label this bond with the symmetry of the corresponding face.

We distinguish between the rotational groups C_n and the dihedral groups D_n which include mirror symmetries. The elements of the group C_n are generated by an element that is also called C_n which is a rotation by angle $\frac{2\pi}{n}$. The elements of the group C_n are thus the identity element E and the rotations $C_n, C_n^2, \ldots, C_n^{n-1}$, which correspond to the rotation angles $0, \frac{2\pi}{n}, \frac{4\pi}{n}, \ldots, \frac{2(n-1)\pi}{n}$. The dihedral group D_n contain the rotations of the group C_n as well as the elements $\sigma, C_n\sigma, C_n^2\sigma, \ldots, C_n^{n-1}\sigma$ which corresponds to the reflection through a mirror axis with orientation angles $0, \frac{\pi}{n}, \frac{2\pi}{n}, \ldots, \frac{(n-1)\pi}{n}$. When the group is a symmetry of a two-dimensional lattice the value of n can only be $1, 2, 3, 4$, and 6 [San93], and we thus have 10 different point groups $C_1, C_2, C_3, C_4, C_6, D_1, D_2, D_3, D_4, D_6$ where C_1 is the trivial group with only the identity element. To check whether a shape has one of these symmetries we go through the following procedure:

1. Find the centroid of the shape and center the shape

2. Search for mirror axis and if found align it with the x-axis

3. For each group we apply all elements of the group and calculate the area of the convex hull of the new vertices generated by this procedure

This procedure yields a symmetry measure given by the area of the original shape divided by the area of the new convex hull. If this symmetry measure is close to 1 the shape has the given symmetry and otherwise it will be smaller than 1. The search for a mirror axis in step 2 is done by computing the principal axis of inertia and align it with the x-axis and check for D_1 symmetry and then align it with the y-axis and check again. In some cases all axes are principal axis of inertia and the procedure is ill-posed. In this case we fall back and check for a mirror axis in all lines going through the centroid and through a vertex or through the midpoint of the line segments. In Figure 3.5 we show an example of a shape with three mirror axes. The original shape is shown in the top left and the remaining shapes are obtained by applying the elements of the point group on the original shape and computing the convex hull of the new points. The other example in Figure 3.6 is a case where the principal axes determination failed and the algorithm falls back to checking every vertex and line segment midpoint for a mirror axis. In this case there is no mirror axis present and the shape has only the symmetry C_3 apart from the trivial C_1.

For the graph neural network application we encode the symmetry as a binary indicator that is 1 when the corresponding symmetry measure is above a certain threshold and use it as an edge feature in the graph.

3.3 Contributions

In the paper *Neural Message Passing with Edge Update Networks for Predicting Properties of Molecules and Materials* in Appendix A we show how the SchNet model [Sch+18] also fits into the message passing framework and further extend the model by adding an edge update network. This architecture yields superior prediction of formation energies and other properties on the tested molecular dataset as well on the tested materials datasets. After the paper was published a more complex model also using edge update networks was proposed by Chen et al. [Che+18]. In addition

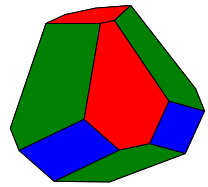


Figure 3.4: Voronoi cell for one atom of Au_2CaSi_2

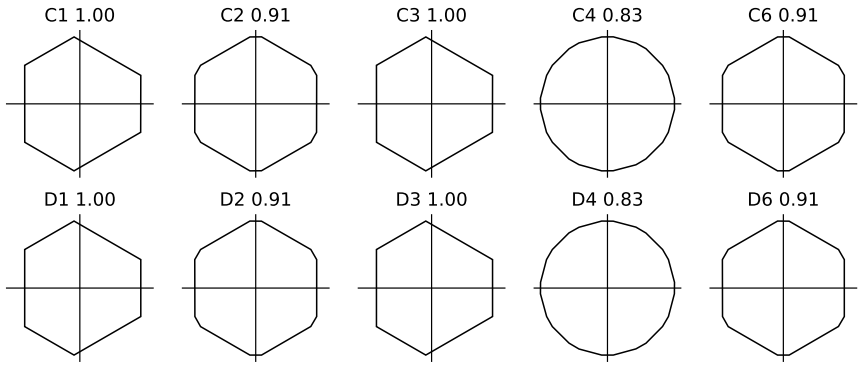


Figure 3.5: Symmetry measures for the shape in the top left. Three mirror axes are present

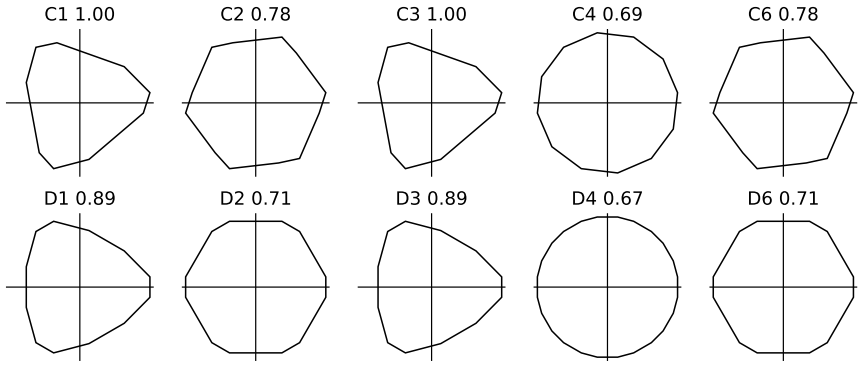


Figure 3.6: Symmetry measures for the shape in the top left. No mirror axis is present

it also incorporates a global state node and set2set readout function and it achieves similar results for the properties and datasets tested.

In the paper *Materials property prediction without atomic positions using graph neural networks* in Appendix B we investigate the efficacy of the model when the exact atom positions are not known. The model then relies only on the unlabeled quotient graph and our local symmetry features as described in the previous chapter. The model achieves a mean absolute error of just 20 meV/atom on the OQMD dataset without access to the distance information. However, when looking at the error for only the unary compositions, the error is a lot higher (108 meV/atom). In the paper we also investigate the mapping between Voronoi graphs and prototypes based on OQMD and we show that in most cases it is possible to map between the two. This means that despite the information loss incurred by the conversion from atomic po-

sitions to unlabeled quotient graph we do not lose the overall structural information described by the prototype. This result is relevant for screening applications, for example when searching for the most stable structure for a given compositions, because in this case one approach would be to test all the relevant prototypes for the lowest energy.

CHAPTER 4

Generative Models

The machine learning models described in the preceding chapters are conditional models. In conditional (or discriminative) modeling we are interested in given a molecule x to predict its property y. Given a dataset of $\{x_i, y_i\}_{i=1}^{N}$ of N molecules and their corresponding properties the goal of a discriminative learning algorithm is to learn the distribution $p(y|x)$. In a generative learning approach we are not only interested in modeling the distribution $p(y|x)$ but also the distribution describing the molecular structure $p(x)$ or the joint distribution $p(y, x)$. The former case is known as unsupervised learning and the latter case is supervised learning.

Referring to the virtual screening funnel in Figure 1.2 the most obvious use of generative models is to also model the generation of new candidate molecules rather than just providing models for the filtering steps, which is where the discriminative models are used. For example in a drug discovery process we would like the generative model to learn what a drug molecule looks like rather than push all possible molecules through the screening funnel. Generative models are also used in semi-supervised learning where only some of the data samples have labels y. Modeling the data manifold of unlabeled data can improve the generalisation performance of the decision boundary between classes as illustrated in Figure 4.1. This is also the case for regression problems where we will often assume that the target property can be modeled as a smooth function on an underlying data manifold.

In this chapter we focus on a particular generative model known as variational

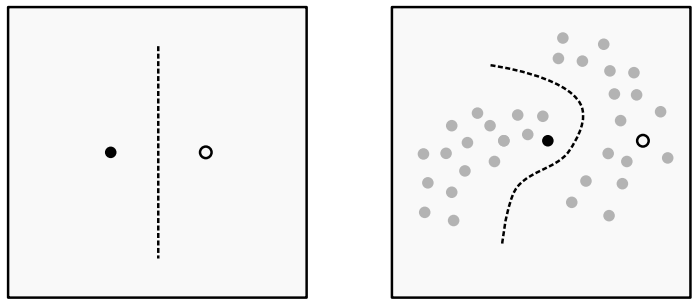


Figure 4.1: The unlabeled data samples (grey) can improve the decision boundary in the classification of the white and black classes

autoencoder. In Section 4.1 we introduce the latent variable model assumption and continue in Section 4.2 with a discussion of the variational autoencoder (VAE), which is a deep learning based inference method for latent variable models. In Section 4.3 we give an overview of generative model trends and applications for molecular sciences and close the chapter with a summary and discussion of our contributions to the field in section 4.4.

4.1 Latent Variable Models

To model a data distribution $p(x)$ we need to capture correlations in the observed variable x. For example in an image we must capture correlations between pixels of the image and for a molecule we need to capture which atom types are likely to appear together. To make it easier to model these correlations we can introduce an unobserved latent variable z and write the marginal distribution as

$$p(x) = \int p(x|z)p(z)dz \tag{4.1}$$

where $p(x|z)$ is the likelihood of the data and $p(z)$ is the prior distribution for the latent variable. Formulating the model in this way can be a lot easier than specifying $p(x)$ directly. For example if we build a generative model for generating pictures of animals z could be the kind of animal and it is easier to specify the rest of the generative model when you know which animal to generate i.e. $p(x|z = \text{"dog"})$. In other words we can sample from the distribution $p(x)$ by first obtaining a sample z^* from the prior $p(z)$ and then sample from the likelihood $p(x|z = z^*)$. Not always will the latent variable have a physical meaning as in this example, but it can be just a convenient way of formulating the model. Often the dimensionality of z will be smaller than that of x. An example of a simple latent variable model is a linear model with additive noise:

$$x = Wz + \varepsilon \tag{4.2}$$

where W is a weight matrix. When the prior $p(z)$ is a Gaussian distribution and the noise is also Gaussian this model is known as probabilistic principal component analysis [Bis06]. In deep latent variable models the simple linear relation between x and z is replaced by a multilayer neural network, i.e.

$$x = f_\theta(z) + \varepsilon, \tag{4.3}$$
$$f_\theta(z) = W_L h_{L-1}(z) + b_L, \tag{4.4}$$
$$h_l(z) = g(W_l h_{l-1}(z)) + b_l, \quad l = 2, \ldots, L-1 \tag{4.5}$$
$$h_1(z) = g(W_1 z + b_1), \tag{4.6}$$

where $g(\cdot)$ is a nonlinear activation function, $W_1, \ldots, W_L$ are weight matrices and $b_1, \ldots, b_L$ are bias vectors. These are trainable parameters and we will refer to the collection of all the trainable parameters as θ.

Given a dataset of N samples $\{x_n\}_{n=1}^N$ we want to maximise (4.1) with respect to the model parameters. In the Gaussian linear model (4.2) it is possible to find a closed form solution to the optimisation problem [TB99; Bis06], but in the nonlinear case we must turn to approximate inference methods, because the integral in (4.1) can in general not be computed in closed form.

4.2 Variational Autoencoder

We want to maximise the marginal distribution with respect to the parameters θ:

$$p_\theta(x) = \int p_\theta(x|z)p(z)dz = \mathbb{E}_{p(z)}\left[p_\theta(x|z)\right] \tag{4.7}$$

Given N data samples we could optimise with respect to the parameters using gradient ascent by approximating the integral with M Monte Carlo samples and computing the gradient:

$$\nabla_\theta \sum_i^N p_\theta(x_i) \approx \nabla_\theta \sum_i^N \frac{1}{M} \sum_j^M p_\theta(x_i|z_j^*) \tag{4.8}$$

where z_j^* is drawn from $p(z)$. The problem with this approach is that it is inefficient when the dimensionality of z is large because many of the samples will fall in an area where $p_\theta(x|z)$ is very small and we therefore need a large number of samples to get an accurate estimate of the integral [Bis06]. To alleviate this problem we can use importance sampling and introduce a proposal distribution $q_\phi(z)$, i.e.

$$p_\theta(x) = \int p_\theta(x|z)p(z)dz \tag{4.9}$$

$$= \int p_\theta(x|z)\frac{q_\phi(z)}{q_\phi(z)}p(z)dz \tag{4.10}$$

$$= \mathbb{E}_{q_\phi(z)}\left[p_\theta(x|z)\frac{p(z)}{q_\phi(z)}\right] \tag{4.11}$$

The optimal proposal distribution is proportional to $p_\theta(x|z)p(z)$ [Owe13; KM53], i.e. $p_\theta(z|x)$. This posterior is intractable and we must use a simpler approximating distribution. We will optimise the approximating distribution to match the posterior and this approach is known as *variational inference* [BKM16]. In traditional variational inference we would choose a family of probability density functions and optimise the parameter ϕ for each data sample i.e. we would have a distribution $q_{\phi_i}(z_i)$ for each x_i in the data set. For very large datasets this can become very slow to optimise. The alternative is to utilise amortised inference [GG14; RMW14], where a neural network is used to output the parameters as a function of the input data sample and thus the approximating distribution is modeled as $q_\phi(z|x)$. Notice that using amortised inference is more restrictive than the free distribution per data sample approach, but

it is potentially much faster and enables sharing of features across the data samples. Amortised inference also enables the interpretation as a variational *autoencoder*, because $q_\phi(z|x)$ *encodes* the input to a latent representation z and $p_\theta(x|z)$ works as the *decoder*.

For numerical stability we take the logarithm of the marginal distribution. The logarithm is monotonically increasing so maximising $\log p(x)$ is equivalent to maximising $p(x)$. Using Jensen's inequality we derive the evidence lower bound (ELBO).

$$\log p_\theta(x) = \log \mathbb{E}_{q_\phi(z|x)} \left[p_\theta(x|z) \frac{p(z)}{q_\phi(z|x)} \right] \tag{4.12}$$

$$\geq \mathbb{E}_{q_\phi(z|x)} \left[\log \left(p_\theta(x|z) \frac{p(z)}{q_\phi(z|x)} \right) \right] \tag{4.13}$$

$$= \mathbb{E}_{q_\phi(z|x)} \left[\log p_\theta(x|z) \right] - \mathbb{E}_{q_\phi(z|x)} \left[\log \frac{q_\phi(z|x)}{p(z)} \right] \tag{4.14}$$

$$= \mathbb{E}_{q_\phi(z|x)} \left[\log p_\theta(x|z) \right] - \mathrm{KL} \left[q_\phi(z|x) || p(z) \right], \tag{4.15}$$

where $\mathrm{KL}\left[q||p\right]$ denotes the Kullback-Leibler divergence from q to p. To see when the bound holds with equality we replace the inequality sign in (4.13) with equality, add a positive constant C on the right hand side and solve for C to see what is missing for the bound to be tight:

$$C = \log p_\theta(x) - \mathbb{E}_{q_\phi(z|x)} \left[\log \left(p_\theta(x|z) \frac{p(z)}{q_\phi(z|x)} \right) \right] \tag{4.16}$$

$$= \mathbb{E}_{q_\phi(z|x)} \left[\log \left(\frac{p_\theta(x) q_\phi(z|x)}{p_\theta(x|z) p(z)} \right) \right] \tag{4.17}$$

$$= \mathbb{E}_{q_\phi(z|x)} \left[\log \left(\frac{q_\phi(z|x)}{p_\theta(z|x)} \right) \right] \tag{4.18}$$

$$= \mathrm{KL} \left[q_\phi(z|x) || p_\theta(z|x) \right] \tag{4.19}$$

where we have used $p_\theta(z|x) = \frac{p_\theta(x|z)p(z)}{p_\theta(x)}$. As we can see, the bound is tight when the approximating distribution is the true posterior.

In a variational autoencoder the first term of (4.15) is often called the reconstruction term and the second term the regularisation term. If we remove the regularisation term the lower bound is the same objective function as used for training a non-probabilistic autoencoder. The regularisation term can in some cases be computed in closed form, for example when both $q_\phi(z|x)$ and $p(z)$ are Gaussian distributions, but in general we need to approximate the expectation using sampling. In most cases using a single sample to approximate the expectation is sufficient. The consensus seems to be that it is better to sample many mini-batches and use a small step size rather than using multiple samples to reduce the variance of the expectation.

As also shown in paper C we can use the bits-back coding [HC93; HV04; Che+16] argument to decompose the bound in a different way. Given the true data distribution

$p_{\text{data}}(x)$ the expectation of the lower bound (4.13) can be written as:

$$\mathbb{E}_{p_{\text{data}}(x)}[L] = \mathbb{E}_{p_{\text{data}}(x)q_\phi(z|x)}\left[\log\left(p_\theta(x|z)\frac{p(z)}{q_\phi(z|x)}\right)\right] \tag{4.20}$$

$$= \mathbb{E}_{p_{\text{data}}(x)q_\phi(z|x)}\left[\log\left(p_\theta(z|x)\frac{p_\theta(x)}{q_\phi(z|x)}\right)\right] \tag{4.21}$$

$$= -\mathrm{H}(p_{\text{data}}) - \mathrm{KL}\left[p_{\text{data}}(x)||p_\theta(x)\right] - \mathbb{E}_{p_{\text{data}}}\left[\mathrm{KL}\left[q_\phi(z|x)||p_\theta(z|x)\right]\right]. \tag{4.22}$$

The first term is the inevitable bound given by the entropy of the data distribution, the second term is the error of the model and the last term is the average error of the approximate posterior distribution. By using an approximate posterior we suffer an extra approximation cost and if the forward model $p_\theta(x|z)$ can model the data without using the latent variable z it is preferable to set the encoder $q_\phi(z|x)$ to the prior distribution $p(z)$ and ignore the latent code. This is problematic if we are interested in the latent representation and several ways to overcome this problem have been proposed. First of all the variational approximation $q_\phi(z|x)$ can not overfit and the use of expressive models can reduce the penalty for using an inexact model. Examples in this direction are normalizing flows [RM15] and hierarchical modeling of the approximate posterior using ladder networks [Ras+15; Søn+16].

Another line of research is arguing that the ELBO (4.15) objective is inappropriate for learning meaningful latent representations and should be adjusted. For example we can further decompose the KL-term in (4.15) as shown in [KM18; HJ16]:

$$\mathbb{E}_{\hat{p}_{\text{data}}}\left[\mathrm{KL}\left[q_\phi(z|x)||p(z)\right]\right] = I_{q_\phi}(x;z) + \mathrm{KL}\left[q_\phi(z)||p(z)\right], \tag{4.23}$$

where $\hat{p}_{\text{data}}(x)$ is the empirical data distribution and $I_{q_\phi}(x;z)$ is the mutual information of x and z with respect to the distribution $q_\phi(x,z) = q_\phi(z|x)\hat{p}_{\text{data}}(x) = \frac{1}{N}\sum_i^N q_\phi(z|x_i)\delta_{x_i}(x)$ and $q_\phi(z)$ is obtained from this distribution by marginalising over x. From this decomposition we can see that the KL-term in the ELBO not only penalises the divergence between the marginal $q_\phi(z)$ and the prior $p(z)$ but also the mutual information between the observed and latent variables. To avoid this effect we can reweight or eliminate these terms in the cost function. See [TBL18] for a survey of different methods using this approach.

Another interpretation of the ELBO objective function is to look at the objective function in terms of a rate-distortion tradeoff [Ale+18; TBL18]:

$$H = \mathbb{E}_{p_{\text{data}}(x)}\left[-\log(p_{\text{data}}(x))\right] \tag{4.24}$$

$$D = \mathbb{E}_{p_{\text{data}}(x)}\left[\mathbb{E}_{q_\phi(z|x)}\left[-\log p_\theta(x|z)\right]\right] \tag{4.25}$$

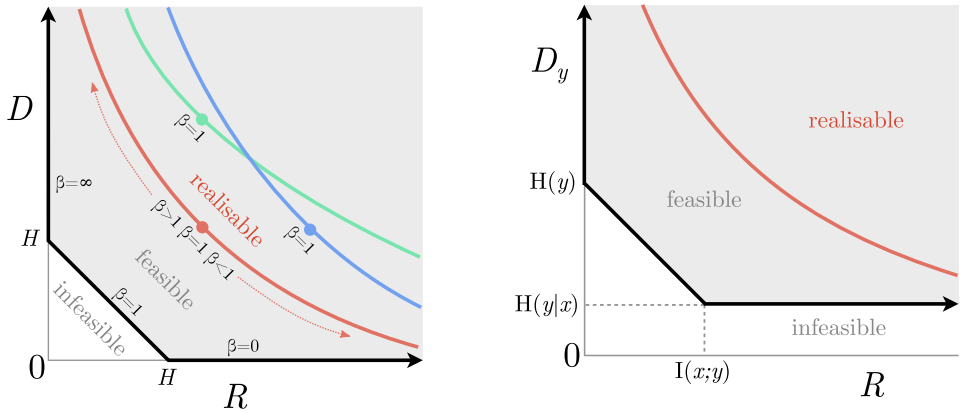
$$R = \mathbb{E}_{p_{\text{data}}(x)}\left[\mathrm{KL}\left[q_\phi(z|x)||m(z)\right]\right] \tag{4.26}$$

where H is the entropy of the data generating distribution, D is the distortion of the VAE measured by the reconstruction negative log-likelihood and R is the rate which is the average KL-divergence between the encoder and a variational approximation $m(z)$

of the marginal $p_e(z) = \mathbb{E}_{p_{\text{data}}(x)}[q_\phi(z|x)]$. The rate can be seen as a measure of how much information is encoded in the latent space. Fixing the variational approximation of the marginal to the prior, i.e $m(z) = p(z)$ the ELBO objective (4.15) can be written as $-(D + R)$ and as $-(D + \beta R)$ for the β-VAE [Hig+17]. It is shown [Ale+18] that different models with different rate and distortion values can have the same value for the ELBO objective and the ELBO is therefore not a good measure for the quality of the model. Furthermore it is shown that the following inequality holds:

$$H - D \leq \mathrm{I}_{p_e(x,z)}(x; z) \leq R, \tag{4.27}$$

where $\mathrm{I}_{p_e(x,z)}(x; z)$ is the mutual information between x and z for the distribution $p_e(x, z) = p_{\text{data}}(x)q_\phi(z|x)$. Equality is achieved in the theoretical limit of perfect encoder and decoder models. This limit is shown as the thick black line in Figure 4.2a. In this limit an optimally trained VAE will land on any point on the line $D = H - R$ and the ELBO will be constant. In practice the model will not be perfect and the rate-distortion tradeoff will fall in the feasible region and we can adjust the tradeoff by tuning $\beta \geq 0$. However, where we land on the curve depends on the model and the quality of the encoder and decoder models and therefore Alemi et al. [Ale+18] suggests to choose a target rate σ and minimise the distortion, i.e. $\min_{\phi,\theta}(D+|\sigma-R|)$. With this approach the model is trained to encode a certain amount of information in the latent space, but as Tschannen, Bachem, and Lucic [TBL18] point out there is no guarantee that the encoded information is useful for the downstream task and what is encoded in the latent representation and what information is stored in the decoder. They go on and define another distortion quantity for supervised learning. For the input x the goal is to learn a latent representation z that is sufficient for predicting



(a) Figure from [Ale+18].

(b) Figure from [TBL18].

Figure 4.2: Rate-distortion tradeoff for (a) unsupervised learning distortion target D and (b) supervised learning distortion target D_y.

y, i.e.

$$D_y = \mathbb{E}_{p_{\text{data}}(x,y)} \left[\mathbb{E}_{q_\phi(z|x)} \left[-\log p_\theta(y|z) \right] \right] \tag{4.28}$$

The rate-distortion tradeoff using this measure of distortion is shown in Figure 4.2b. In the best case scenario the latent representation encodes all the mutual information $\mathrm{I}(x, y)$ between x and y at the rate $\mathrm{I}(x, y)$ and the minimum achievable distortion is given by $\mathrm{H}(x|y)$. Again this point is only achievable with perfect encoder and decoder models and in practice the performance of the model will fall elsewhere in the feasible region. These unsupervised and supervised information theoretical views of representation learning have been unified by Alemi and Fischer [AF18] and their treatment also generalises to the semi-supervised scenario.

4.3 Generating new Molecules

When building a generator for molecules we are interested in the following properties:

- Realism - The molecules should be chemically valid and resemble real world molecules.
- Diversity - The generator is used for exploration and should therefore be able to generate unseen molecules, i.e. it needs to generalise beyond the data used for training.
- Usefulness - Often we are interested in molecules with certain properties, e.g. molecules that are easy to synthesize, drug-like molecules or solar cell materials with a specific band gap.

As we will discuss in the following section there already exists a myriad of generative models for molecules. However, even though we know that we are seeking models that fulfills the three properties above, it is hard to quantitatively evaluate the quality of the different methods because there are no definitive measures for realism, diversity and usefulness. Therefore, two research groups have in parallel developed benchmarks for molecule generation [Pol+18; Bro+18]. One way of measuring realism is to extract different properties of a reference dataset and the generated dataset such as number of atoms, number of aromatic rings etc. The realism score is then how closely these distribution match. Diversity can be quantified by identifying how large a percentage of the generated dataset does not already exist in the reference data. The usefulness is very application specific but could for example be the drug-likeness score [Bic+12].

Generating new molecules have traditionally been done through hand-coded rules or genetic algorithms (e.g [BP94; Kam+02; SBB15]), where new molecules are crafted based on a library of substructures. Lately the focus has shifted towards deep learning models, primarily based on VAEs [KPH17; Dai+17; JBJ18; Liu+18], adversarial autoencoders [Kad+17b; Kad+17a; Bla+17], generative adversarial networks (GANs) [Gui+17; DK18] and reinforcement learning [Jaq+16; Oli+17; Seg+18; PIT18]. A schematic view of the four types of models is shown in Figure 4.3. As discussed above

the variational autoencoder minimises reconstruction distortion and encoder rate. In a generative adversarial network (GAN) [Goo+14] a generator gets a noise sample as input and generates a sample that looks like the real data. Another neural network (the discriminator) is trained to distinguish between a sample from the real dataset and a sample from the generator. Thus the generator and discriminator are playing a game against each other. For image generation GANs have produced remarkably realistic images of faces [KLA18], arguably more realistic than results based on VAEs. The vector input of a GAN can also be regarded as a latent space, but unlike VAEs there is no readily available encoder that maps from input/output space to the latent space.

The adversarial autoencoder (AAE) [Mak+15] combines the idea of GANs with that of VAEs. As in VAEs an encoder-decoder architecture is trained but instead of using the VAE KL-term to regularise the encoder a discriminator network is trained to distinguish between samples from the encoder and samples from a prior. The idea is that it allows the use of arbitrarily complex priors and that the loss imposed by the discriminator may be a more suitable measure of divergence between the two distributions than the KL-divergence [Mak+15].

In a reinforcement learning framework we can think of a molecule as being generated by a sequence of actions performed by an agent. At the end of the generation process the agent receives a reward or penalty depending on the properties of

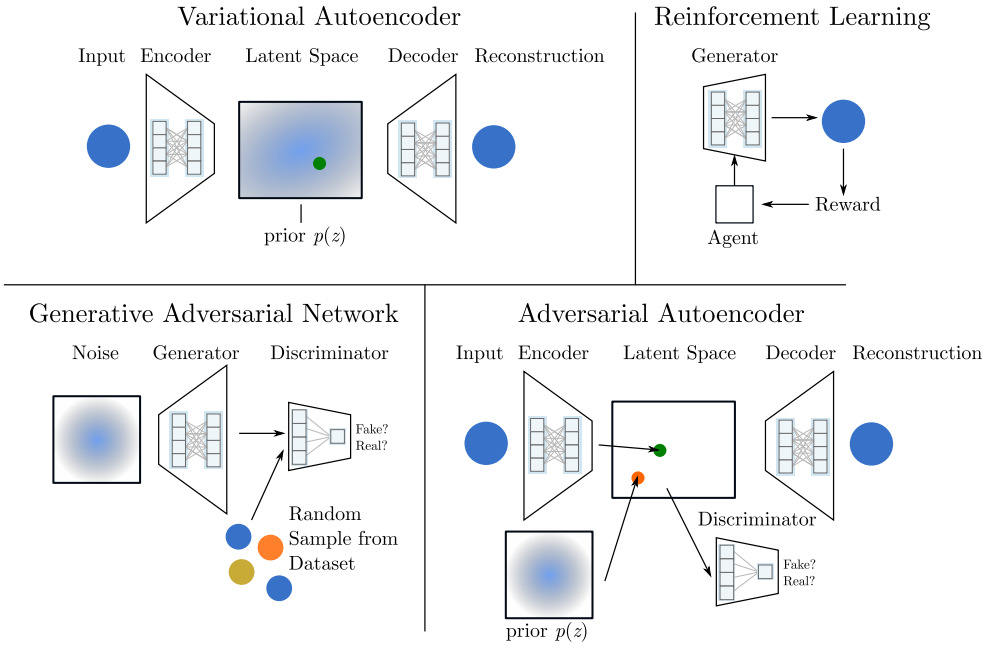


Figure 4.3: Generative deep learning models used for generating molecules.

the generated molecule. Thus this framework enables optimisation of the generated structures towards a desired property and does not require the property to be a differentiable function of the actions taken by the agent. In ORGAN [Gui+17] and MolGAN [DK18] reinforcement learning is combined with a GAN, i.e. part of the reward is whether the generator can fool the discriminator into believing the generated molecule comes from the training distribution.

Finally it is also possible to train a deep learning based generator in isolation, either conditioned on a target property or unconditionally. These models are often autoregressive and generate a molecule sequentially, e.g. [Li+18; You+18b; LZL18; Seg+18; GGS18].

4.3.1 Representations in Generative Models

How to generate a molecule depends on the choice of representation. Conventional fingerprints such as MACCS or ECFP [RH10] are in general not suitable for this task because they are most often not unique in the sense that different molecules map to the same fingerprint. Most applications can be categorized into one of the following types:

- Sequential generation of strings, such as SMILES and InChI.
- One-shot molecular graph generation based on adjacency matrix.
- Sequential molecular graph generation.
- Sequential placement of atoms in three dimensional space.

The first use of variational autoencoder for generating SMILES strings was demonstrated by Gómez-Bombarelli et al. [Góm+16] using a similar architecture as used for the first recurrent neural network (RNN) VAE based sentence generator [Bow+15]. One of the downsides of character-based generation of molecules is that it would often generate invalid strings, either syntactically invalid or semantically invalid SMILES strings. The problem of generating syntactically invalid strings has been eliminated by GrammarVAE [KPH17] by encoding the SMILES strings as a sequence of formal grammar production rules and enforcing the grammar on the output of the RNN based VAE decoder. In defence of character-based generators it can be argued whether generating a few invalid strings is problematic, because they can easily be discarded using a syntax checker. Furthermore if the GrammarVAE decoder assigns a lot of probability mass to production rules that are afterwards masked out by the enforcement of the grammar the generator has not really learned what the correct syntax is.

One-shot molecular graph generation for small graphs using the adjacency matrix was first proposed by Simonovsky and Komodakis [SK18] and uses a VAE to generate the adjacency matrix. To compute the reconstruction loss a graph matching problem must be solved which does not scale well to larger graphs. By using a GAN architecture the reconstruction loss does not need to be calculated and this approach was

taken by De Cao and Kipf [DK18]. The generator still needs to decide on a specific node ordering, but it is free to choose one that is suitable for the task.

In sequential generation of graphs, nodes and edges are placed sequentially. A common theme in recently proposed models [Li+18; Liu+18; You+18b; You+18a; Ass+18] is that placement of nodes and edges are done in an autoregressive fashion, i.e. conditioned on the currently generated intermediate graph. The current global graph state is obtained using a graph neural network. This type of architecture was first proposed by Johnson [Joh17] for generating intermediate graph representation of relational information extracted from natural language. In the models proposed by Li et al. [Li+18] and You et al. [You+18b] only the generative process is modeled and the generation process alternates between adding nodes and adding edges to the most recently placed node. The addition of a new node/edge depends on the whole sequence of generator steps, which can make the models hard to train. In the VAE based models by Liu et al. [Liu+18] and Assouel et al. [Ass+18] all the nodes are created first based on the encoded latent representation and then edges are added between the nodes. In the model proposed by Liu et al. [Liu+18] adding edges depends only on the intermediate graph and not on the generation history, which makes training easier. The edge generation by Assouel et al. [Ass+18] is fully differentiable, because it is only dependent on the node hidden states and not on the other edges in the graph. Finally, instead of generating just a graph structure Gebauer, Gastegger, and Schütt [GGS18] propose a model for autoregressive generation of molecules in three dimensions. In each step an atom is placed and a neural network outputs a probability density function over the distance between the newly placed atom and each of the already placed atoms. The atom is then placed by sampling from this distribution.

4.3.2 Inverse Design

So far we have discussed different architectures for generating molecules, but in many cases we are interested in generating molecules with a given property. This is known as inverse design in the molecular science literature. From an engineering perspective this is a misnomer, because it is a natural design process to choose a set of requirements and then design a device (in this case a molecule) to fulfill those requirements. However it seems that the name has stuck because the conventional approach is to start out from a given material and try to optimise it towards a desired target property. The models described above use one of the following approaches:

- Reinforcement learning
- Conditional generator
- Latent space optimisation

The three approaches are illustrated in Figure 4.4.

We have already discussed reinforcement learning models, which is one way of guiding a generator towards a given target by providing a reward signal. One example

is the sequential molecular graph generation with reinforcement learning proposed by You et al. [You+18a]. A disadvantage of reinforcement learning is that the generator can collapse to a single data point because the generative process is trained as part of the optimisation process. The diversity of the generator can be increased by including a discriminator loss in the reward function. The discriminator will quickly learn that the single generated data point is the "fake", forcing the generator to increase the diversity.

In the conditional generator approach the generator is trained with labeled data points and instead of modeling just the distribution of unlabeled data $p(x)$ we try to learn the conditional distribution $p(x|y)$, where y is the target label. When generating new samples we can then condition on a target value y^* and generate samples from $p(x|y = y^*)$. This approach requires labeled data to be available.

In a VAE or GAN the decoder/generator is also a conditional generator, but it is conditioned on the latent variable z, i.e. the decoder is a model for $p(x|z)$. After training the VAE we can train a separate model $p_\theta(y|z)$ on the latent representation to minimise D_y in (4.28). Once this model has been obtained we search the latent space for a latent representation z^* that maximises the conditional density, i.e.

$$z^* = \underset{z}{\operatorname{argmax}}\, p_\theta(y = y^*|z) \tag{4.29}$$

and then generate molecules from the decoder conditioned on this latent representation $p(x|z = z^*)$. This is well suited for semi-supervised learning where all data samples are used for training the VAE and only the labeled data is used for training $p_\theta(y|z)$. Latent space optimisation will not work if the latent space is bad for predicting the property of interest as discussed in Section 4.2. Furthermore, the optimisation problem in (4.29) may be difficult to solve if $p_\theta(y|z)$ is a complicated model and regularisation may be needed to avoid areas of the latent space where either $p_\theta(y|z)$ or $p_\theta(x|z)$ is poor.

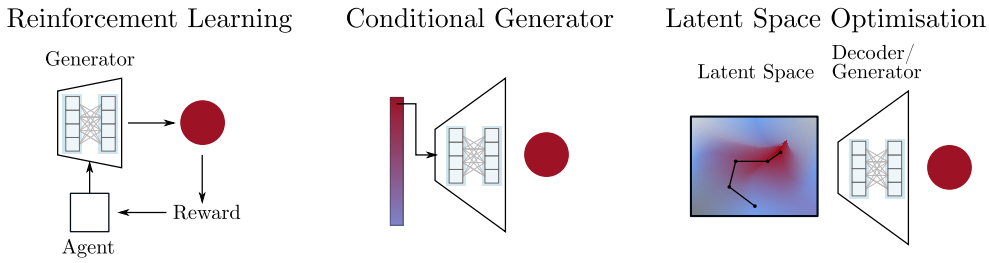


Figure 4.4: Generative models used for inverse design

4.4 Contributions

In the preceding section we have given a short overview of generative models for molecules. This thesis include two papers on this topic.

In the paper *Deep Generative Models for Molecular Science* in Appendix C we give a short review of deep generative models for molecular science with focus on VAEs and tricks of the trade to train them. In this paper we do not discuss the rate-distortion tradeoff, because this perspective was not available when the paper was written.

In the paper *Machine learning-based screening of complex molecules for polymer solar cells* in Appendix D we build upon GrammarVAE [KPH17] and design an application specific grammar instead of SMILES for representing molecules for polymer solar cells. The molecules consists of building blocks from a library of acceptor and donor molecules as well as side groups and by using a tailored grammar the representation is much more compact than SMILES. We utilise the β-VAE for training, because we noticed that without down-weighting the KL-term the latent representation is ignored. A neural network is trained on the latent space representation to predict the HOMO/LUMO energies (highest occupied molecular orbital and lowest unoccupied molecular orbital, respectively). To circumvent the problems of latent space optimisation discussed above we instead sample from modes of $p_\theta(x, z)$ and score the samples based on the predicted HOMO/LUMO energies. The best 100 molecules were selected for DFT calculations and 61% was within the desired target tolerance compared to just 11% for the randomised search used for gathering the training set.

CHAPTER 5

Conclusion

Deep learning methods for screening of molecules and materials are still in an early phase. In the preceding chapters we have introduced the two main topics of the PhD project: 1) prediction of properties of molecules and materials using graph neural networks and 2) generative models for molecules.

Graphs are arguably a natural representation for molecules and materials and the challenge lies in designing algorithms that can learn from this representation. The recent development of graph neural network models show a lot of potential in learning from large databases of DFT calculations by exploiting the relational nature of the data. Graph neural networks such as the SchNet model [Sch+18] shares some similarities with the atom-centered symmetry functions (ACSF) by Behler [Beh11]. In both cases a neural network is used to map each atom's chemical environment to an energy contribution. The key difference is that in the former case the representation of the atom's environment is also learned by a neural network model while for the ACSF representation the chemical environment representation is given by a set of predefined symmetry functions. In our paper *Neural Message Passing with Edge Update Networks for Predicting Properties of Molecules and Materials* we propose a message passing model that can be seen as an extension of the SchNet model [Sch+18]. Instead of only learning a representation for atoms and their local environments the model also learns a latent representation for the bonds between atoms and this added flexibility leads to better prediction accuracy. The model yields highly accurate energy predictions of molecules and materials based on large publicly available datasets with mean absolute errors on the order of 11 meV for QM9 [Rud+12], 23 meV/atom for Materials Project [Jai+13] and 15 meV/atom for OQMD [Saa+13; Kir+15]. Furthermore our results shows that the overall accuracy is not sensitive to the choice of cutoff distance as long as the distance is large enough and using just the k nearest neighbours or the Voronoi neighbours work equally well. However, these results are based on knowing the atomic positions of the molecules and spatial structure of the materials and for screening purposes we do normally not have this information before doing relaxation with DFT calculations. This problem is explored in the paper *Materials property prediction without atomic positions using graph neural networks* and we show that by using the model developed in the first publication we can also predict the energy of materials just from knowing the connectivity. We propose a new method for encoding local rotational symmetry into the graph which improves the predictive accuracy in comparison to the setting where only the connectivity is known.

The test error on OQMD is 20 meV/atom when including symmetry information and 26 meV/atom without symmetry information. However, it is also noted that even with relatively large databases such as OQMD only a very small area of the chemical space is covered which can make extrapolation to materials outside OQMD difficult. As demonstrated on our $ABSe_3$ dataset this problem is alleviated by pretraining on OQMD which drastically reduces the number of training samples needed to make accurate predictions on the new dataset.

In generative models for molecules the goal is to not just model the mapping from molecules to property but to also model the distribution of molecules. This can be used to generate new molecules that are similar to a given set of molecules. We have focused on latent variable models and more specifically Variational Autoencoders (VAEs) which implies we also learn a latent vector space representation for the molecules. As discussed in the previous chapter it can be challenging to learn a useful latent representation because the commonly used ELBO objective function does not directly reflect the inherent rate-distortion-usefulness tradeoff. Different tricks to train VAEs have been proposed in the literature and we give an overview of these tricks and applications in molecular science in the paper *Deep Generative Models for Molecular Science* in Appendix C. In the paper *Machine learning-based screening of complex molecules for polymer solar cells* in Appendix D we demonstrate the use of a VAE-based generative model to accelerate the discovery of new molecules suitable for polymer solar cells by doing optimisation in the learned latent space.

5.1 Open Questions and Future Directions

The graph neural networks are very expressive models and are able to fit complicated mappings. They generalise well, which may be partly attributed to the inductive relational bias. However, with the high number of parameters the model requires a relatively large amount of training data to perform well. The datasets we have used for training consist of only equilibrium structures and they are fitted to a single target label for each sample. However, with such expressive models and with higher computational budget it might be possible to learn from non-equilibrium structures as well and train the model to predict energy, forces or even the electron density. By using this training data that is currently discarded future models may be able to learn the underlying physics to a larger degree and therefore generalise better.

The regression models presented in this thesis give only point estimates of target properties, i.e. the uncertainties of the predictions are not modeled. The inherent uncertainty in the DFT calculations could be modeled directly by letting the model output a mean and a variance, but a much more challenging problem is to model uncertainty when the model is extrapolating to an area of the chemical space that it has no knowledge of. Future research should go into developing methods for incorporating this kind of uncertainty into the model. Some pointers to pursue could be Bayesian neural networks that assigns priors to the weights of a neural network, but often these priors are hard to set in a meaningful way. Additionally, the models can

easily become intractable without very rough approximations. Other possible candidates are noise contrastive priors [Haf+18] or ensemble methods with randomised prior functions [OAC18]. In the former case the model is trained to predict high uncertainty for noisy versions of the training samples, but it is not straightforward to choose the right amount of noise and it is not given that the method generalises well for high-dimensional data because the noisy data is still close to the training data. In ensemble methods several models are trained on the same datasets and different techniques are used to make the models different from each other. This can be achieved by training on different subsets of the data [LPB17], regularising the weights by biasing them towards their initial values [Pea+18] or adding a hand-crafted random function to each ensemble member [OAC18]. Further development of these methods for specific applications in molecules and materials screening could lead to better uncertainty estimates than those offered by current models.

Good uncertainty estimates in combination with generative models would be able to automatise the whole screening process of new molecules and materials. The models need to be further refined towards active learning where a model is capable of scheduling new experiments or more precise quantum mechanical computations if the uncertainty is high on a promising path in the chemical space. The chemical space can be modeled using latent variable models, but the generative models have some room for improvement. The learned latent representation is not always appropriate for the task and perhaps this can be improved by directly modeling the inherent rate-distortion-usefulness tradeoff in the training cost function.

Bibliography

[AF18] Alexander A Alemi and Ian Fischer. "TherML: Thermodynamics of Machine Learning". In: (July 2018). arXiv: 1807.04162 [cs.LG].

[Ale+18] Alexander Alemi et al. "Fixing a Broken ELBO". In: *Proceedings of the 35th International Conference on Machine Learning*. Edited by Jennifer Dy and Andreas Krause. Volume 80. Proceedings of Machine Learning Research. Stockholmsmässan, Stockholm Sweden: PMLR, 2018, pages 159–168.

[Ass+18] Rim Assouel et al. "DEFactor: Differentiable Edge Factorization-based Probabilistic Graph Generation". In: (November 2018). arXiv: 1811.09766 [cs.LG].

[Bat+16] Peter W Battaglia et al. "Interaction Networks for Learning about Objects, Relations and Physics". In: (December 2016). arXiv: 1612.00222 [cs.AI].

[Bat+18] Peter W Battaglia et al. "Relational inductive biases, deep learning, and graph networks". In: (June 2018). arXiv: 1806.01261 [cs.LG].

[Beh11] Jörg Behler. "Atom-centered symmetry functions for constructing high-dimensional neural network potentials". en. In: *J. Chem. Phys.* 134.7 (February 2011), page 074106.

[Beh16] Jörg Behler. "Perspective: Machine learning potentials for atomistic simulations". en. In: *J. Chem. Phys.* 145.17 (November 2016), page 170901.

[Bic+12] G Richard Bickerton et al. "Quantifying the chemical beauty of drugs". en. In: *Nat. Chem.* 4.2 (January 2012), pages 90–98.

[Bis06] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006. ISBN: 0387310738.

[BKC13] Albert P Bartók, Risi Kondor, and Gábor Csányi. "On representing chemical environments". In: *Phys. Rev. B Condens. Matter* 87.18 (May 2013), page 184115.

[BKM16] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. "Variational Inference: A Review for Statisticians". In: (January 2016). arXiv: 1601.00670 [stat.CO].

[Bla+17] Thomas Blaschke et al. “Application of generative autoencoder in de novo molecular design”. In: (November 2017). arXiv: 1711.07839 [cs.LG].

[Bow+15] Samuel R Bowman et al. “Generating Sentences from a Continuous Space”. In: (November 2015). arXiv: 1511.06349 [cs.LG].

[BP94] Tilman Brodmeier and Ern Pretsch. “Application of genetic algorithms in molecular modeling”. In: *J. Comput. Chem.* 15.6 (June 1994), pages 588–595.

[Bro+17] Felix Brockherde et al. “Bypassing the Kohn-Sham equations with machine learning”. In: *Nat. Commun.* 8.1 (October 2017), page 872.

[Bro+18] Nathan Brown et al. “GuacaMol: Benchmarking Models for De Novo Molecular Design”. In: (November 2018). arXiv: 1811.09621 [q-bio.QM].

[Bru+13] Joan Bruna et al. “Spectral Networks and Locally Connected Networks on Graphs”. In: (December 2013). arXiv: 1312.6203 [cs.LG].

[Che+16] Xi Chen et al. “Variational Lossy Autoencoder”. In: (November 2016). arXiv: 1611.02731 [cs.LG].

[Che+18] Chi Chen et al. “Graph Networks as a Universal Machine Learning Framework for Molecules and Crystals”. In: (December 2018). arXiv: 1812.05055 [cond-mat.mtrl-sci].

[CHK84] S J Chung, Th Hahn, and W E Klee. “Nomenclature and generation of three-periodic nets: the vector method”. In: *Acta Crystallogr. A* 40.1 (January 1984), pages 42–50.

[Col+18] Christopher R Collins et al. “Constant size descriptors for accurate machine learning models of molecular properties”. en. In: *J. Chem. Phys.* 148.24 (June 2018), page 241718.

[Cyb89] G Cybenko. “Approximation by superpositions of a sigmoidal function”. In: *Math. Control Signals Systems* 2.4 (December 1989), pages 303–314.

[Dai+17] H Dai et al. “Syntax-Directed Variational Autoencoder for Molecule Generation”. In: *NIPS 2017 Workshop Machine Learning for Molecules and Materials*. Long Beach, California, 2017.

[De+16] Sandip De et al. “Comparing molecules and solids across structural and alchemical space”. en. In: *Phys. Chem. Chem. Phys.* 18.20 (April 2016), pages 13754–13769.

[Del+17] Olaf Delgado-Friedrichs et al. “Crystal structures as periodic graphs: the topological genome and graph databases”. In: *Struct. Chem.* 28.1 (February 2017), pages 39–44.

[DK18] Nicola De Cao and Thomas Kipf. “MolGAN: An implicit generative model for small molecular graphs”. In: (May 2018). arXiv: 1805.11973 [stat.ML].

[DM07] M. De Graef and M.E. McHenry. *Structure of Materials: An Introduction to Crystallography, Diffraction and Symmetry*. Cambridge University Press, 2007. ISBN: 9780521651516.

[DO03] Olaf Delgado-Friedrichs and Michael O'Keeffe. "Identification of and symmetry computation for crystal nets". In: *Acta Crystallogr. A* 59.4 (July 2003), pages 351–360.

[Duv+15] David Duvenaud et al. "Convolutional Networks on Graphs for Learning Molecular Fingerprints". In: (September 2015). arXiv: `1509.09292 [cs.LG]`.

[Eon11] Jean-Guillaume Eon. "Euclidian embeddings of periodic nets: definition of a topologically induced complete set of geometric descriptors for crystal structures". In: *Acta Crystallogr. A* 67.1 (January 2011), pages 68–86.

[Eon98] Jean-Guillaume Eon. "Geometrical Relationships between Nets Mapped on Isomorphic Quotient Graphs: Examples". In: *J. Solid State Chem.* 138.1 (June 1998), pages 55–65.

[Fab+15] Felix Faber et al. "Crystal structure representations for machine learning models of formation energies". In: *Int. J. Quantum Chem.* 115.16 (August 2015), pages 1094–1101.

[Fab+17] Felix A Faber et al. "Prediction Errors of Molecular Machine Learning Models Lower than Hybrid DFT Error". en. In: *J. Chem. Theory Comput.* 13.11 (November 2017), pages 5255–5264.

[Foo+18] David Fooshee et al. "Deep learning for chemical reaction prediction". en. In: *Mol. Syst. Des. Eng.* 3.3 (June 2018), pages 442–452.

[GG14] Samuel Gershman and Noah Goodman. "Amortized inference in probabilistic reasoning". In: *Proceedings of the annual meeting of the cognitive science society*. Volume 36. 2014.

[GGS18] Niklas W A Gebauer, Michael Gastegger, and Kristof T Schütt. "Generating equilibrium molecules with deep neural networks". In: (October 2018). arXiv: `1810.11347 [stat.ML]`.

[Gil+17] Justin Gilmer et al. "Neural Message Passing for Quantum Chemistry". In: (April 2017). arXiv: `1704.01212 [cs.LG]`.

[GMS05] 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.* Volume 2. July 2005, 729–734 vol. 2.

[Góm+16] Rafael Gómez-Bombarelli et al. "Automatic chemical design using a data-driven continuous representation of molecules". In: (October 2016). arXiv: `1610.02415 [cs.LG]`.

[Goo+14] Ian Goodfellow et al. “Generative Adversarial Nets”. In: *Advances in Neural Information Processing Systems 27*. Edited by Z Ghahramani et al. Curran Associates, Inc., 2014, pages 2672–2680.

[Gri+10] Thomas L Griffiths et al. “Probabilistic models of cognition: exploring representations and inductive biases”. en. In: *Trends Cogn. Sci.* 14.8 (August 2010), pages 357–364.

[Gui+17] Gabriel Lima Guimaraes et al. “Objective-Reinforced Generative Adversarial Networks (ORGAN) for Sequence Generation Models”. In: (May 2017). arXiv: `1705.10843 [stat.ML]`.

[Haf+18] Danijar Hafner et al. “Reliable Uncertainty Estimates in Deep Neural Networks using Noise Contrastive Priors”. In: (July 2018). arXiv: `1807.09289 [stat.ML]`.

[Han+15] Katja Hansen et al. “Machine Learning Predictions of Molecular Properties: Accurate Many-Body Potentials and Nonlocality in Chemical Space”. en. In: *J. Phys. Chem. Lett.* 6.12 (June 2015), pages 2326–2331.

[HC93] Geoffrey E Hinton and Drew van Camp. “Keeping the Neural Networks Simple by Minimizing the Description Length of the Weights”. In: *Proceedings of the Sixth Annual Conference on Computational Learning Theory*. COLT ’93. Santa Cruz, California, USA: ACM, 1993, pages 5–13.

[He+15] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: (December 2015). arXiv: `1512.03385 [cs.CV]`.

[Hel+15] Stephen R Heller et al. “InChI, the IUPAC International Chemical Identifier”. en. In: *J. Cheminform.* 7 (May 2015), page 23.

[Hig+17] Irina Higgins et al. “ -VAE: LEARNING BASIC VISUAL CONCEPTS WITH A CONSTRAINED VARIATIONAL FRAMEWORK”. In: *International Conference on Learning Representations (ICLR)*. 2017.

[HJ16] Matthew D Hoffman and Matthew J Johnson. “ELBO surgery: yet another way to carve up the variational evidence lower bound”. In: Barcelona, 2016.

[HL16] Bing Huang and O Anatole von Lilienfeld. “Communication: Understanding molecular representations in machine learning: The role of uniqueness and target similarity”. en. In: *J. Chem. Phys.* 145.16 (October 2016), page 161102.

[HL17] Bing Huang and O Anatole von Lilienfeld. “The “DNA” of chemistry: Scalable quantum machine learning with “amons””. In: (July 2017). arXiv: `1707.04146 [physics.chem-ph]`.

[Hor91] Kurt Hornik. “Approximation capabilities of multilayer feedforward networks”. In: *Neural Netw.* 4.2 (January 1991), pages 251–257.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-Term Memory”. In: *Neural Comput.* 9.8 (November 1997), pages 1735–1780.

[HSW89] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. “Multilayer feedforward networks are universal approximators”. In: *Neural Netw.* 2.5 (January 1989), pages 359–366.

[HV04] Antti Honkela and Harri Valpola. “Variational learning and bits-back coding: an information-theoretic view to Bayesian learning”. en. In: *IEEE Trans. Neural Netw.* 15.4 (July 2004), pages 800–810.

[Hy+18] Truong Son Hy et al. “Predicting molecular properties with covariant compositional networks”. en. In: *J. Chem. Phys.* 148.24 (June 2018), page 241745.

[Jai+13] Anubhav Jain et al. “Commentary: The Materials Project: A materials genome approach to accelerating materials innovation”. In: *APL Materials* 1.1 (July 2013), page 011002.

[Jaq+16] Natasha Jaques et al. “Sequence Tutor: Conservative Fine-Tuning of Sequence Generation Models with KL-control”. In: (November 2016). arXiv: `1611.02796 [cs.LG]`.

[JB18] Ankit Jain and Thomas Bligaard. “Atomic-position independent descriptor for machine learning of material properties”. In: *Phys. Rev. B Condens. Matter* 98.21 (December 2018), page 214112.

[JBJ18] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. “Junction Tree Variational Autoencoder for Molecular Graph Generation”. In: (February 2018). arXiv: `1802.04364 [cs.LG]`.

[Joh17] Daniel D Johnson. “Learning Graphical State Transitions”. In: *International Conference on Learning Representations (ICLR)*. 2017.

[Kad+17a] Artur Kadurin et al. “druGAN: An Advanced Generative Adversarial Autoencoder Model for de Novo Generation of New Molecules with Desired Molecular Properties in Silico”. en. In: *Mol. Pharm.* 14.9 (September 2017), pages 3098–3104.

[Kad+17b] Artur Kadurin et al. “The cornucopia of meaningful leads: Applying deep adversarial autoencoders for new molecule development in oncology”. en. In: *Oncotarget* 8.7 (February 2017), pages 10883–10890.

[Kam+02] Stefan Kamphausen et al. “Genetic algorithm for the design of molecules with desired properties”. en. In: *J. Comput. Aided Mol. Des.* 16.8-9 (August 2002), pages 551–567.

[KB11] Matthew A Kayala and Pierre F Baldi. “A Machine Learning Approach to Predict Chemical Reactions”. In: *Advances in Neural Information Processing Systems 24*. Edited by J Shawe-Taylor et al. Curran Associates, Inc., 2011, pages 747–755.

[Kea+16] Steven Kearnes et al. “Molecular graph convolutions: moving beyond fingerprints”. en. In: *J. Comput. Aided Mol. Des.* 30.8 (August 2016), pages 595–608.

[Kir+15] Scott Kirklin et al. “The Open Quantum Materials Database (OQMD): assessing the accuracy of DFT formation energies”. en. In: *npj Computational Materials* 1 (December 2015), page 15010.

[KLA18] Tero Karras, Samuli Laine, and Timo Aila. “A Style-Based Generator Architecture for Generative Adversarial Networks”. In: (December 2018). arXiv: `1812.04948 [cs.NE]`.

[Kle04] W E Klee. “Crystallographic nets and their quotient graphs”. In: *Cryst. Res. Technol.* 39.11 (November 2004), pages 959–968.

[KM18] Hyunjik Kim and Andriy Mnih. “Disentangling by Factorising”. In: (February 2018). arXiv: `1802.05983 [stat.ML]`.

[KM53] H Kahn and A W Marshall. “METHODS OF REDUCING SAMPLE SIZE IN MONTE CARLO COMPUTATIONS”. In: *Journal of the Operations Research Society of America* 1.5 (November 1953), pages 263–278.

[Kon+18] Risi Kondor et al. “Covariant Compositional Networks For Learning Graphs”. February 2018.

[KPH17] Matt J Kusner, Brooks Paige, and José Miguel Hernández-Lobato. “Grammar Variational Autoencoder”. en. In: *International Conference on Machine Learning*. July 2017, pages 1945–1954.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “ImageNet Classification with Deep Convolutional Neural Networks”. In: *Advances in Neural Information Processing Systems 25*. Edited by F Pereira et al. Curran Associates, Inc., 2012, pages 1097–1105.

[Lec85] Yann Lecun. “Une procedure d’apprentissage pour reseau a seuil asymmetrique (A learning scheme for asymmetric threshold networks)”. English (US). In: *Proceedings of Cognitiva 85, Paris, France*. 1985, pages 599–604.

[LHS15] Emanuel A Lazar, Jian Han, and David J Srolovitz. “Topological framework for local structure analysis in condensed matter”. en. In: *Proc. Natl. Acad. Sci. U. S. A.* 112.43 (October 2015), E5769–76.

[Li+15] Yujia Li et al. “Gated Graph Sequence Neural Networks”. In: (November 2015). arXiv: `1511.05493 [cs.LG]`.

[Li+18] Yujia Li et al. “Learning Deep Generative Models of Graphs”. In: (March 2018). arXiv: `1803.03324 [cs.LG]`.

[Liu+17] Qin Liu et al. “Improving the Performance of Long-Range-Corrected Exchange-Correlation Functional with an Embedded Neural Network”. en. In: *J. Phys. Chem. A* 121.38 (September 2017), pages 7273–7281.

[Liu+18] Qi Liu et al. “Constrained Graph Variational Autoencoders for Molecule Design”. In: *Advances in Neural Information Processing Systems 31*. Edited by S Bengio et al. Curran Associates, Inc., 2018, pages 7806–7815.

[LJ18] Hongzhou Lin and Stefanie Jegelka. “ResNet with one-neuron hidden layers is a Universal Approximator”. In: *Advances in Neural Information Processing Systems 31*. Edited by S. Bengio et al. Curran Associates, Inc., 2018, pages 6172–6181. URL: `http://papers.nips.cc/paper/7855-resnet-with-one-neuron-hidden-layers-is-a-universal-approximator.pdf`.

[LPB17] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. “Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles”. In: *Advances in Neural Information Processing Systems 30*. Edited by I Guyon et al. Curran Associates, Inc., 2017, pages 6402–6413.

[LZL18] Yibo Li, Liangren Zhang, and Zhenming Liu. “Multi-Objective De Novo Drug Design with Conditional Graph Generative Model”. In: (January 2018). arXiv: `1801.07299 [q-bio.QM]`.

[Mak+15] Alireza Makhzani et al. “Adversarial Autoencoders”. In: (November 2015). arXiv: `1511.05644 [cs.LG]`.

[Mal+13] Alex Malins et al. “Identification of structure in condensed matter with the topological cluster classification”. en. In: *J. Chem. Phys.* 139.23 (December 2013), page 234506.

[Mer+14] B Meredig et al. “Combinatorial screening for new materials in unconstrained composition space with machine learning”. In: *Phys. Rev. B Condens. Matter* 89.9 (March 2014), page 094104.

[Mon+12] Grégoire Montavon et al. “Learning Invariant Representations of Molecules for Atomization Energy Prediction”. In: *Advances in Neural Information Processing Systems 25*. Edited by F Pereira et al. Curran Associates, Inc., 2012, pages 440–448.

[Nag+18] Ryo Nagai et al. “Neural-network Kohn-Sham exchange-correlation potential and its out-of-training transferability”. en. In: *J. Chem. Phys.* 148.24 (June 2018), page 241737.

[OAC18] Ian Osband, John Aslanides, and Albin Cassirer. “Randomized Prior Functions for Deep Reinforcement Learning”. In: (June 2018). arXiv: `1806.03335 [stat.ML]`.

[Oli+17] Marcus Olivecrona et al. “Molecular De Novo Design through Deep Reinforcement Learning”. In: (April 2017). arXiv: `1704.07555 [cs.AI]`.

[Owe13] Art B. Owen. *Monte Carlo theory, methods and examples*. 2013.

[Pea+18] Tim Pearce et al. “Bayesian Inference with Anchored Ensembles of Neural Networks, and Application to Reinforcement Learning”. In: (May 2018). arXiv: `1805.11324 [stat.ML]`.

[PIT18] Mariya Popova, Olexandr Isayev, and Alexander Tropsha. “Deep reinforcement learning for de novo drug design”. en. In: *Sci Adv* 4.7 (July 2018), eaap7885.

[Pol+18] Daniil Polykovskiy et al. “Molecular Sets (MOSES): A Benchmarking Platform for Molecular Generation Models”. In: (November 2018). arXiv: 1811.12823 [cs.LG].

[Ram+14] Raghunathan Ramakrishnan et al. “Quantum chemistry structures and properties of 134 kilo molecules”. en. In: *Sci Data* 1 (August 2014), page 140022.

[Ran01] M Randić. “Novel shape descriptors for molecular graphs”. en. In: *J. Chem. Inf. Comput. Sci.* 41.3 (May 2001), pages 607–613.

[Ras+15] Antti Rasmus et al. “Semi-Supervised Learning with Ladder Networks”. In: (July 2015). arXiv: 1507.02672 [cs.NE].

[RH10] David Rogers and Mathew Hahn. “Extended-connectivity fingerprints”. en. In: *J. Chem. Inf. Model.* 50.5 (May 2010), pages 742–754.

[RHW86] D E Rumelhart, G E Hinton, and R J Williams. “Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1”. In: edited by David E Rumelhart, James L McClelland, and Corporate PDP Research Group. Cambridge, MA, USA: MIT Press, 1986. Chapter Learning Internal Representations by Error Propagation, pages 318–362.

[RLB18] Matthias Rupp, O Anatole von Lilienfeld, and Kieron Burke. “Guest Editorial: Special Topic on Data-Enabled Theoretical Chemistry”. en. In: *J. Chem. Phys.* 148.24 (June 2018), page 241401.

[RM15] Danilo Jimenez Rezende and Shakir Mohamed. “Variational Inference with Normalizing Flows”. In: (May 2015). arXiv: 1505.05770 [stat.ML].

[RMW14] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. “Stochastic Backpropagation and Approximate Inference in Deep Generative Models”. In: *Proceedings of The 31st International Conference on Machine Learning*. 2014, pages 1278–1286.

[Ros62] Frank Rosenblatt. *Principles of neurodynamics; perceptrons and the theory of brain mechanisms*. Washington: Spartan Books, 1962.

[RST18] Kevin Ryczko, David Strubbe, and Isaac Tamblyn. “Deep Learning and Density Functional Theory”. In: (November 2018). arXiv: 1811.08928 [cond-mat.mtrl-sci].

[Rud+12] Lars Ruddigkeit et al. “Enumeration of 166 billion organic small molecules in the chemical universe database GDB-17”. en. In: *J. Chem. Inf. Model.* 52.11 (November 2012), pages 2864–2875.

[Rup+12] Matthias Rupp et al. “Fast and accurate modeling of molecular atomization energies with machine learning”. en. In: *Phys. Rev. Lett.* 108.5 (February 2012), page 058301.

[Saa+13] James E Saal et al. “Materials Design and Discovery with High-Throughput Density Functional Theory: The Open Quantum Materials Database (OQMD)”. en. In: *JOM* 65.11 (November 2013), pages 1501–1509.

[San93] D.E. Sands. *Introduction to Crystallography*. Dover Books on Chemistry Series. Dover Publications, 1993. ISBN: 9780486678399. URL: `https://books.google.dk/books?id=h%5C_A5u5sczJoC`.

[SBB15] Adriana Supady, Volker Blum, and Carsten Baldauf. “First-Principles Molecular Structure Search with a Genetic Algorithm”. en. In: *J. Chem. Inf. Model.* 55.11 (November 2015), pages 2338–2348.

[Sca+09] Franco Scarselli et al. “The graph neural network model”. en. In: *IEEE Trans. Neural Netw.* 20.1 (January 2009), pages 61–80.

[Sch+14] K T Schütt et al. “How to represent crystal structures for machine learning: Towards fast prediction of electronic properties”. In: *Phys. Rev. B Condens. Matter* 89.20 (May 2014), page 205118.

[Sch+17a] Kristof T Schütt et al. “Quantum-chemical insights from deep tensor neural networks”. en. In: *Nat. Commun.* 8 (January 2017), page 13890.

[Sch+17b] Philippe Schwaller et al. ““Found in Translation”: Predicting Outcomes of Complex Organic Chemistry Reactions using Neural Sequence-to-Sequence Models”. In: (November 2017). arXiv: `1711.04810 [cs.LG]`.

[Sch+18] K T Schütt et al. “SchNet - A deep learning architecture for molecules and materials”. en. In: *J. Chem. Phys.* 148.24 (June 2018), page 241722.

[Seg+18] Marwin H S Segler et al. “Generating Focused Molecule Libraries for Drug Discovery with Recurrent Neural Networks”. en. In: *ACS Cent Sci* 4.1 (January 2018), pages 120–131.

[SK18] Martin Simonovsky and Nikos Komodakis. “GraphVAE: Towards Generation of Small Graphs Using Variational Autoencoders”. In: (February 2018). arXiv: `1802.03480 [cs.LG]`.

[Søn+16] Casper Kaae Sønderby et al. “Ladder Variational Autoencoders”. In: (February 2016). arXiv: `1602.02282 [stat.ML]`.

[TB99] Michael E Tipping and Christopher M Bishop. “Probabilistic Principal Component Analysis”. In: *J. R. Stat. Soc. Series B Stat. Methodol.* 61.3 (1999), pages 611–622.

[TBL18] Michael Tschannen, Olivier Bachem, and Mario Lucic. “Recent Advances in Autoencoder-Based Representation Learning”. In: (December 2018). arXiv: `1812.05069 [cs.LG]`.

[Thi09] Georg Thimm. “Crystal topologies–the achievable and inevitable symmetries”. en. In: *Acta Crystallogr. A* 65.Pt 3 (May 2009), pages 213–226.

[Tho+18] Nathaniel Thomas et al. “Tensor field networks: Rotation- and translation-equivariant neural networks for 3D point clouds”. In: (February 2018). arXiv: 1802.08219 [cs.LG].

[VBK15] Oriol Vinyals, Samy Bengio, and Manjunath Kudlur. “Order Matters: Sequence to sequence for sets”. In: (November 2015). arXiv: 1511.06391 [stat.ML].

[War+17] Logan Ward et al. “Including crystal structure attributes in machine learning models of formation energies via Voronoi tessellations”. In: *Phys. Rev. B Condens. Matter* 96.2 (July 2017), page 024104.

[Wei88] David Weininger. “SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules”. In: *J. Chem. Inf. Comput. Sci.* 28.1 (February 1988), pages 31–36.

[You+18a] Jiaxuan You et al. “Graph Convolutional Policy Network for Goal-Directed Molecular Graph Generation”. In: *Advances in Neural Information Processing Systems 31*. Edited by S Bengio et al. Curran Associates, Inc., 2018, pages 6410–6421.

[You+18b] Jiaxuan You et al. “GraphRNN: A Deep Generative Model for Graphs”. In: (February 2018). arXiv: 1802.08773 [cs.LG].

APPENDIX A

Neural Message Passing with Edge Update Networks for Predicting Properties of Molecules and Materials

The following paper is published as,

Peter Bjørn Jørgensen, Karsten Wedel Jacobsen, and Mikkel N. Schmidt. 2018. "Neural Message Passing with Edge Updates for Predicting Properties of Molecules and Materials." arXiv [stat.ML]. arXiv:1806.03146. Poster presentation at Machine Learning for Molecules and Materials workshop at NeurIPS 2018.

arXiv:1806.03146v1 [stat.ML] 8 Jun 2018

Neural Message Passing with Edge Updates for Predicting Properties of Molecules and Materials

Peter Bjørn Jørgensen
Department of Applied Mathematics and Computer Science
Technical University of Denmark
pbjo@dtu.dk

Karsten Wedel Jacobsen
Department of Physics
Technical University of Denmark
kwj@fysik.dtu.dk

Mikkel N. Schmidt
Department of Applied Mathematics and Computer Science
Technical University of Denmark
mnsc@dtu.dk

Abstract

Neural message passing on molecular graphs is one of the most promising methods for predicting formation energy and other properties of molecules and materials. In this work we extend the neural message passing model with an edge update network which allows the information exchanged between atoms to depend on the hidden state of the receiving atom. We benchmark the proposed model on three publicly available datasets (QM9, The Materials Project and OQMD) and show that the proposed model yields superior prediction of formation energies and other properties on all three datasets in comparison with the best published results. Furthermore we investigate different methods for constructing the graph used to represent crystalline structures and we find that using a graph based on K-nearest neighbors achieves better prediction accuracy than using maximum distance cutoff or the Voronoi tessellation graph.

1 Introduction

The current workhorse for screening of new molecules and materials is Density Functional Theory (DFT), but machine learning methods show a potential for trading DFT accuracy for a several orders of magnitude decrease in computation time. Until recently machine learning methods for predicting properties of molecules and materials have been based on hand crafted feature descriptors, such as Coulomb matrix (Rupp et al., 2012), bag-of-bonds (Hansen et al., 2015), fingerprints (e.g. Rogers and Hahn 2010) and histogram of angles (Faber et al., 2017). With the availability of large databases of molecules and materials we are now seeing a shift towards data-driven representation learning as we have seen in the computer vision field.

The graph neural network model was introduced by Gori et al. (2005); Scarselli et al. (2009) and regained attention when Li et al. (2015) expanded the model for different graph problems. In the last few years a number of graph-based models for molecules have been proposed (Duvenaud et al., 2015;

Preprint.

Kearnes et al., 2016; Schütt et al., 2017a,b; Gilmer et al., 2017). These models can all be cast into the framework of message passing on a molecular graph as shown by Gilmer et al. (2017).

The molecular graphs used as input for these models are either the topological graph defined by the chemical bonds (with or without bond lengths) or a fully connected graph where the pairwise distances between all the atoms are used as edge features. The topological approach does not directly apply to crystalline structures because the chemical bonds are less well-defined and the fully connected approach is impossible because the structure is infinite. The first application of neural message passing for materials is SchNet (Schütt et al., 2017c) where a constant cutoff distance is used. As noted by Schütt et al. (2017c) this may lead to "isolated atoms" if the cutoff distance is too small and the computational burden increases with the size of the cutoff. This motivates us to investigate different ways to define the molecular graphs based on nearest neighbor cutoff. Concurrently with the work presented in this paper Xie and Grossman (2018) have also employed neural message passing on graphs based on Voronoi tessellation and K-nearest neighbors, but they do not include a comparison between the methods.

The edge neural network with set-to-set readout function (enn-s2s) (Gilmer et al., 2017) demonstrated state of the art prediction accuracy on the 13 properties of the QM9 (Ramakrishnan et al., 2014; Ruddigkeit et al., 2012) dataset consisting of 134k molecules. The recently proposed SchNet (Schütt et al., 2017b,c) network improves the accuracy on eight out of the twelve properties. In both of these models the information exchanged between the atoms in the message passing scheme depends on the representation of the sending atom and the edge feature on which the message is passed, but is independent of the representation of the receiving atom. Gilmer et al. (2017) proposed the use of a "pair message" network that includes the state of the receiving atom, but its predicting performance is inferior to the enn-s2s network. In this work we propose to extend the SchNet model with an edge update network such that the edge feature depends on the representation of the atoms that the edge connects. This in turn means the information exchanged between the atoms also depends on the receiving atom. Edge update networks was also utilised in the Weave module proposed by Kearnes et al. (2016), but there the edges are forced to be undirected and the prediction accuracy is below that of enn-s2s (Gilmer et al., 2017).

We benchmark the proposed edge update network on QM9 molecules, Materials Project (Jain et al., 2013) and OQMD (Saal et al., 2013; Kirklin et al., 2015) datasets and it shows an improvement over current state of the art results on all three datasets. For crystalline structures we show that it is beneficial to use a K-nearest neighbor graph rather than a graph defined by a constant cutoff distance as used in previous work (Schütt et al., 2017c).

The paper is organised as follows. We present the proposed model within the neural message passing framework in Section 2 and we show how it is different from other models within the same framework in Section 3. We introduce the three datasets in Section 4, which are used for benchmarking the model in Section 5 and we conclude the paper in Section 6.

2 Message Passing Neural Networks

We describe message passing neural networks similarly to (Gilmer et al., 2017) as a model operating on a graph G with vertex features x_v and edge features ε_{vw}. Each vertex has a hidden state h_v^t and each edge has a hidden state e_{vw}^t which are updated in a number of interaction steps T. Vertices are updated using a message function $M_t(\cdot)$ and a state transition function $S_t(\cdot)$

$$m_v^{t+1} = \sum_{w \in N(v)} M_t(h_v^t, h_w^t, e_{vw}^t), \tag{1}$$

$$h_v^{t+1} = S_t\left(h_v^t, m_v^{t+1}\right), \tag{2}$$

where $N(v)$ denotes the neighbourhood of v, i.e. the vertices that have an edge to v. The edges are updated by an edge update function $E_t(\cdot)$ that depends of the previous edge state and the states of the sending and receiving vertices of the current step

$$e_{vw}^{t+1} = E_t\left(h_v^{t+1}, h_w^{t+1}, e_{vw}^t\right). \tag{3}$$

After the T interaction steps a readout function $R(\cdot)$ is applied which maps the set of vertex states into a single entity

$$\hat{y} = R\left(\{h_v^T \in G\}\right). \tag{4}$$

The readout function must be invariant to permutation of the vertex set, which is often achieved via summation over the vertex features. The functions M_t, S_t, E_t and R are all implemented as neural networks with trainable parameters and can be optimised using gradient descent. Some models, for example Deep Tensor Neural Network (Schütt et al., 2017a), use weight sharing across the interaction steps, thus $M_t \equiv M$, $S_t \equiv S$ and $E_t \equiv E$ for all time steps t. A range of graph convolution models (Duvenaud et al., 2015; Li et al., 2015; Battaglia et al., 2016; Kearnes et al., 2016; Schütt et al., 2017a) including Laplacian based models (Bruna et al., 2013) can be cast into this message passing framework as shown by Gilmer et al. (2017).

2.1 SchNet with Edge Update Network

We now describe how our model fits into the message passing framework described above. As vertex input features, x_v, we use the atomic numbers which are translated into an embedding vector for each atomic number as in (Schütt et al., 2017a,b,c). The initial hidden state $h_v^0 = \ell(x_v)$ is thus the result of a lookup function $\ell : \mathbb{Z} \to \mathbb{R}^C$. The hidden state is a representation of the atom and its chemical environment and the idea behind the message passing interaction steps is to refine this representation based on the surrounding atoms and their chemical environments. Since the interaction between atoms depends on the distance between them (Coulomb's law) we use the interatomic distances as initial edge features. We then expand the initial edge feature using a radial basis function. Denoting the distance between atom v and w as d_{vw}, the edge between node v and w in the graph has initial feature vector

$$(\varepsilon_{vw})_k = \exp\left(-\frac{(d_{vw} - (-\mu_{\text{min}} + k\Delta))^2}{\Delta}\right), k = 0 \dots k_{\text{max}} \tag{5}$$

where μ_{min}, Δ, and k_{max} are chosen such that the centers of the functions covers the range of the input features. In all our experiments we set $\mu_{\text{min}} = 0\,\text{Å}, \Delta = 0.1\,\text{Å}, k_{\text{max}} = 150$. This expansion makes it easier for the neural network model to decorrelate input and output similar to how 1-hot-encoding is preferred over integer coding for categorical features.

The role of the hidden edge representation e_{vw}^t is to control how the two connected atoms interact. The idea of using an edge update network is to let the updated atomic hidden states influence this interaction. We use an edge update function at each interaction step implemented as a two layer feed-forward neural network. The input to the network is the concatenation of the edge representation and the hidden states of the receiving and sending vertices. Thus

$$e_{vw}^0 = E_0(h_v^0, h_w^0, \varepsilon_{vw}) = g(W_{\text{E2}}^0 g(W_{\text{E1}}^0(h_v^0; h_w^0; \varepsilon_{vw}))), \tag{6}$$

and similarly for the subsequent steps

$$e_{vw}^{t+1} = g(W_{\text{E2}}^{t+1} g(W_{\text{E1}}^{t+1}(h_v^{t+1}; h_w^{t+1}; e_{vw}^t))), \tag{7}$$

where $g(x) = \ln(\mathrm{e}^x + 1) - \ln(2)$ is the shifted soft-plus activation function $(\cdot;\cdot)$ denotes vector concatenation and $\{W_{\text{E1}}^0, W_{\text{E2}}^0, W_{\text{E1}}^{t+1}, W_{\text{E2}}^{t+1}\}$ are trainable weight matrices. This update makes the network edges directional.

The message function itself is only a function of the sending node and can be written as

$$M_t(h_v^t, h_w^t, e_{vw}^t) = M_t(h_w^t, e_{vw}^t) = (W_1^t h_w^t) \odot g(W_3^t g(W_2^t e_{vw}^t)), \tag{8}$$

where $\odot$ denotes element-wise multiplication and $\{W_1^t, W_2^t, W_3^t\}$ are trainable weight matrices. This message function enables the interpretation of the function on the right hand side of the element-wise multiplication as a filter-generating function (Schütt et al., 2017b,c) $f_g^t(e_{vw}^t) = g(W_3^t g(W_2^t e_{vw}^t))$, a continuous analogous to the filters applied over a discrete domain in a convolutional neural network for image processing. We visualize this function towards the end of this section. The shifted soft-plus activation function is chosen to follow Schütt et al. (2017b,c) and can be seen as an infinitely differentiable alternative to the rectified linear unit (ReLU) activation function.

The state transition function applies a two layer neural network on the sum of incoming messages and adds this to the current hidden state as in Residual Networks (He et al., 2015):

$$S_t\left(h_v^t, m_v^{t+1}\right) = h_v^t + W_5^t g(W_4^t m_v^{t+1}), \tag{9}$$

where $\{W_4^t, W_5^t\}$ are trainable weight matrices. After a number of interaction steps T all the information about the property we want to predict must be contained in the set of hidden node states.

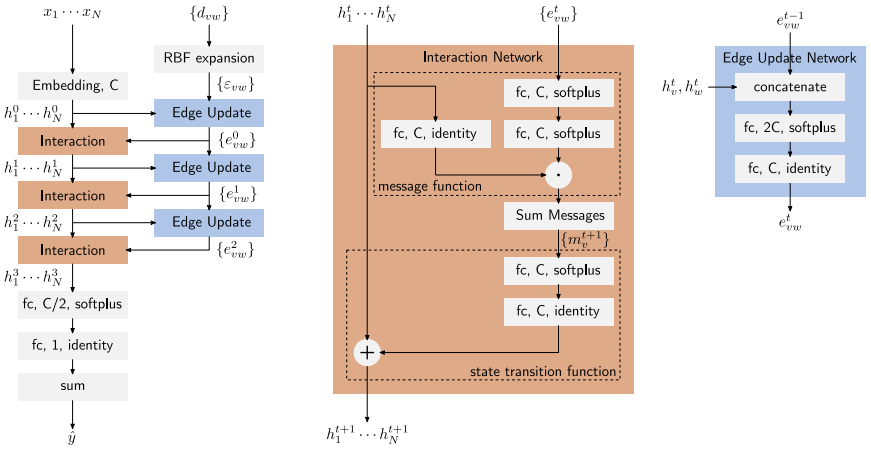


Figure 1: Flow of computations of the proposed model. The dimension of the hidden state for each atom state is C and the "fc"-blocks are fully connected layers annotated with the output dimension and the applied activation function.

We apply a readout function for which we use a two layer neural network that maps the hidden representation to a scalar and finally we sum over the contribution from each atom, i.e.

$$R\left(\{h_v^T \in G\}\right) = \sum_{h_v^T \in G} W_7 g(W_6 h_v^T), \tag{10}$$

where $\{W_7, W_6\}$ are also trainable weight matrices. The model architecture and flow of computations is illustrated in Figure 1. The original SchNet architecture is obtained if we "remove" the edge update function, i.e. by setting $E_t(h_v^t, h_w^t, e_{vw}^{t-1}) = e_{vw}^{t-1}$. This property enables direct comparison between the two models.

We want to visualize the filter generating function to qualitatively assess what the model learns, but because of the edge update network, the edge feature of the first layer e_{vw}^0 does not only depend on the distance d_{vw} between the two atoms, but also on the atom embeddings, i.e.

$$f_g^0(d_{vw}) = f_g^0(E_0(h_v^0, h_w^0, \mathrm{RBF}(d_{vw}))), \tag{11}$$

where $\mathrm{RBF}(\cdot)$ is the radial basis function expansion defined in (5). We can now plot the filter response of the first layer as a function of the sending and receiving atoms and the distance between them as shown in Figure 2 and Figure 3. We see that the filter-generating function is almost identical across the sending atom species but the variation across the receiving atoms is more significant. This indicates that the model has learned to shape the messages depending on the receiving atom. The ability to condition the filter on the pair of sending and receiving atoms is the key difference between the proposed model and the SchNet model which is using the same filter for all combinations of sending and receiving atoms.

3 Related Work

Our model is closely related to the other message passing neural network models used for molecular properties prediction. However, the only message passing neural network that has explored the use of edge updates is the Weave module (Kearnes et al., 2016). The message function is $M_t(h_v^t, h_w^t, e_{vw}^t) = \alpha(W_1^t e_{vw}^t)$ and the state transition function is $S_t(h_v^t, m_v^{t+1}) = \alpha(W_3^t(\alpha(W_2^t h_v^t); m_v^{t+1}))$ where $\alpha(\cdot)$ is the rectified linear unit activation function, $(\cdot;\cdot)$ denotes vector concatenation and $\{W_1^t, W_2^t, W_3^t\}$ are weight matrices. Unlike our model where the edge updates are interleaved with the node updates, the edge updates are done in parallel, i.e.

$$\begin{aligned} e_{vw}^{t+1} &= E_t\left(h_v^t, h_w^t, e_{vw}^t\right) \\ &= \alpha(W_6^t(\alpha(W_4^t(h_v, h_w)) + \alpha(W_4^t(h_w, h_v)), \alpha(W_5^t e_{vw}^t))) \end{aligned} \tag{12}$$

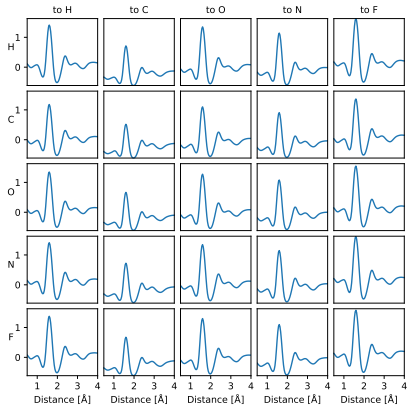


Figure 2: An example of one of the learned filters (out of 64) at the first layer of the message passing architecture trained on prediction of formation energy U_0 of the QM9 dataset. The filter depends on the embedding of the sending (the rows) and the receiving (the columns) nodes.

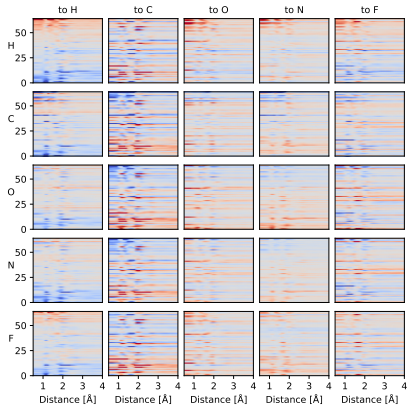


Figure 3: Filter variation for all 64 learned filters at the first layer of the message passing architecture trained on prediction of formation energy U_0 of the QM9 dataset. Each row of pixels corresponds to one of the 64 filters and the color encodes the deviation (-0.4 is blue and 0.4 is red) from the corresponding average filter (averaged over sending and receiving atom). The enumeration of the filters is arbitrary and we have sorted them according to the value of the deviation at 1.0 for the filter H→H. The filter generating function shows a higher dependence on the species of the receiving atom than the species of the sending atom.

Another difference is that the edge updates in the Weave module are invariant to permutation of h_v and h_w and the edges are thus undirected by design. The Weave module does not use the Euclidean distance between the atoms as input features, but uses the graph distance on the chemical graph. The model was tested on a range of different classification tasks as well as drug efficacy, photovoltaic efficiency and solubility regression tasks. The model was reimplemented by Faber et al. (2017) and Gilmer et al. (2017). The version by Faber et al. (2017) includes a few modifications. The edges are no longer forced to be undirected, i.e.

$$\begin{aligned} e_{vw}^{t+1} &= E_t\left(h_v^t, h_w^t, e_{vw}^t\right) \\ &= \alpha(W_6^t(\alpha(W_4^t(h_v, h_w)), \alpha(W_5^t e_{vw}^t))) \end{aligned} \tag{13}$$

and the Euclidean distance between the atoms is included in the message function by division of the edge feature with a range of different powers of the distance d_{vw}:

$$M_t(h_v^t, h_w^t, e_{vw}^t) = \alpha\left(\underset{k\in\{0,1,2,3,6\}}{\text{concatenate}}\left(\frac{W_1^t e_{vw}^t}{d_{vw}^k}\right)\right) \tag{14}$$

Both versions of the graph convolution model based on the Weave module shows similar performance on the QM9 regression benchmark (Gilmer et al., 2017; Faber et al., 2017), but the mean absolute prediction error is significantly higher than that of the enn-s2s and SchNet models.

4 Datasets

We benchmark the proposed model on three publicly available datasets.

Quantum Machines 9 (QM9) (Ramakrishnan et al., 2014; Ruddigkeit et al., 2012) The dataset contains 133 885 examples of stable small organic molecules with up to 7 heavy atoms (CONF) and

Table 1: Target properties of the QM9 dataset.

Target	Description
$\varepsilon_{\text{HOMO}}$	Energy of highest occupied molecular orbital (HOMO)
$\varepsilon_{\text{LUMO}}$	Energy of lowest occupied molecular orbital (LUMO)
$\Delta\varepsilon$	Difference between LUMO and HOMO
ZPVE	Zero point vibrational energy
μ	Dipole moment
α	Isotropic polarizability
$\langle R^2 \rangle$	Electronic spatial extent
U_0	Internal energy at 0 K
U	Internal energy at 298.15 K
H	Enthalpy at 298.15 K
G	Free energy at 298.15 K
C_v	Heat capacity at 298.15 K

up to 29 atoms in total including H. The 12 target properties for each example are shown in Table 1. All properties are calculated at the B3LYP/6-31G(2df,p) level of quantum chemistry. Following Gilmer et al. (2017); Schütt et al. (2017c) we randomly select a training set of 110 000 molecules, a validation set of 10 000 molecules and the remaining 13 885 examples are used for testing. The validation set is used for early stopping and model selection.

The Materials Project (Jain et al., 2013) This dataset contains geometries and formation energies of 69 640 inorganic compounds with input structures taken from the ICSD database (Bergerhoff et al., 1983). We use the latest version of the database (version 2.0.0). The target property is heat of formation. The number of examples is reduced to 69 539 after we exclude all materials with noble gases (He, Ne, Ar, Kr, Xe) because they occur so infrequently in the dataset. This brings the number of different elements in the dataset down to 84. Following (Schütt et al., 2017c) we randomly sample 60 000 of the examples to use as the training set. Of the remaining examples we use a set of 5000 for validation and the remaining 4539 examples are used for testing.

Open Quantum Materials Database (OQMD) (Saal et al., 2013; Kirklin et al., 2015) Is also a database of inorganic structures. We have extracted the database from the supplemental material of (Ward et al., 2017). Again we consider materials with noble gases as outliers and we also exclude (highly unstable) materials with a heat of formation of more than $5\,\text{eV}/\text{atom}$. We thus exclude 210 out of 435 792 examples. As with the materials project we use 20% (87 116) of the examples as a test set, 5000 for validation and the remaining 343 466 for training.

5 Results

We train the models with ADAM (Kingma and Ba, 2014) with initial learning rate 5×10^{-4} or 1×10^{-4} (when the higher learning rate leads to instability) and decrease the learning rate by multiplying with 0.96 every 100 000 gradient steps. We use a minibatch size of 32, run the optimization for up to 1×10^7 gradient steps, compute the validation error every 50 000 steps and terminate if there was no improvement within the last 1×10^6 steps. When applying the model on the two materials datasets (Materials Project and OQMD) we use an average over messages rather than a sum in (1) because Schütt et al. (2017c) found that it increases the stability of the optimization. For these datasets (as well as for some of the QM9 properties, specifically $\varepsilon_{\text{HOMO}}$, $\varepsilon_{\text{LUMO}}$, $\Delta\varepsilon$, ZPVE, μ, α and $\langle R^2 \rangle$) we also take the average rather than the sum in the readout function (10) because the target property is formation energy *per atom*. When the readout function uses a sum we first estimate the target mean and standard deviation atom-wise, i.e. $\hat{\mu} = \sum_i \frac{t_i}{n_i}, \hat{\sigma} = \sqrt{\sum_i (\frac{t_i}{n_i} - \hat{\mu})^2}$ where t_i and n_i are the target and number of atoms for the ith training sample. The targets are then normalised using $\tilde{t_i} = \frac{t_i - n_i \hat{\mu}}{\hat{\sigma}}$ such that each scalar of the sum (10) is expected to have zero mean and unit variance for a given sample with known n_i. When the readout function uses the average rather than a sum the targets are normalised to zero mean and unit variance in the "usual" way, irrespectively of the number of atoms. For the QM9 experiments we use a hidden state representation of dimension $C = 64$ and

Table 2: Mean absolute error of formation energy predictions for V-RF, SchNet and the proposed model. For QM9 the error is in meV and for the Materials Project and OQMD the numbers are in $\mathrm{meV/atom}$. The lowest error is highlighted in bold. We have obtained the V-RF results by running the implementation provided by the authors (Ward et al., 2017), while SchNet results have been obtained by running our own SchNet implementation. The numbers in parenthesis are the estimated 95th percentile, which have been obtained by sampling the test set (with replacement) 1×10^6 times.

Model	QM9 U_0	Mat.Proj.	OQMD
V-RF	-	76.8 (79.8)	74.5 (75.1)
SchNet	13.6 (14.2)	31.8 (33.3)	27.5 (27.9)
Proposed Model	**10.5** (11.1)	**22.7** (24.0)	**14.9** (15.2)

for the two materials datasets we increase the dimensionality to $C = 256$ because the number of different elements in these datasets is 84 rather than 5 in QM9. In agreement with the results shown in (Schütt et al., 2017c) and also noticed in (Gilmer et al., 2017) we did not see a gain in prediction accuracy when using more than 3 interaction steps, so the results presented here are are all with $T = 3$.

5.1 Edge Update Network

To assess the effect of the edge update network we train the proposed model with and without the edge update network to predict formation energies of the three datasets. Without the edge update network the model reduces to the SchNet model. For the two materials datasets (Materials Project and OQMD) we use a cutoff distance of 5 Å as used in (Schütt et al., 2017c) when constructing the graphs. In the benchmark we also include a state of the art non-deep learning, graph-based method, which creates a graph based on the Voronoi diagram, extracts a number of features from the graph and uses a random forest regression model for making predictions and we refer to this method as V-RF (Ward et al., 2017). The model also uses the spatial information, but the interatomic distances are normalised such that the model's predictions are independent of the scaling. We benchmark all three models on the same training and test set, but V-RF also uses the validation set for training because early stopping is not used for this model. The mean absolute error for the test set predictions are shown in Table 2. In all three benchmarks we observe a big improvement in prediction accuracy when using the edge update network. This is not only due to the increase in number of parameters, because merely increasing the number of interaction steps does not improve the performance of the model.

We also train the proposed model to predict the 12 properties of the QM9 dataset. The results of SchNet (Schütt et al., 2017c) and enn-s2s (Gilmer et al., 2017) are included as references. The results are shown in Table 3. The proposed model demonstrates a significant improvement in the prediction of 9 out of 12 properties and matches the existing results for the remaining 3.

5.2 Choosing The Cutoff

We want to investigate the importance of choosing a cutoff when constructing the graph used as input to the neural message passing models. The choice of cutoff is important because the computational complexity of the algorithm increases linearly with the number of edges in the graph. On the other hand if the number of edges is too small the interaction between the nodes of the graph may be too limited. In (Schütt et al., 2017c) and in our experiments above a constant cutoff distance of 5 Å is used. We use the OQMD dataset for this experiment and use the formation energy as the target for predictions. With this cutoff distance some of the atoms of the dataset becomes isolated and further increasing the cutoff distance comes with prohibitive computational cost. Alternatively we can use a K-nearest cutoff method such that each atom receives messages from the K nearest neighboring atoms no matter how far away they are. This makes the connections between atoms asymmetrical, but that is not necessarily a problem. Finally we can use the neighbors as obtained through Voronoi tessellation, i.e. two atoms are connected if they are neighboring cells in a Voronoi diagram. The connections are symmetrical and we also avoid isolated atoms. We use the software package Voro++ (Rycroft, 2009) to compute the tessellation.

Table 3: Mean absolute error of predictions for different target properties of the QM9 dataset using 110k training examples. The lowest error is highlighted in bold. SchNet and enn-s2s results are from (Schütt et al., 2017c) and (Gilmer et al., 2017) respectively. The numbers in parenthesis are the estimated 95th percentile, which have been obtained by sampling the test set (with replacement) 1×10^6 times.

Target	Unit	SchNet	enn-s2s	Proposed	(95th)
$\varepsilon_{\text{HOMO}}$	meV	41	43	**36.7**	(37.3)
$\varepsilon_{\text{LUMO}}$	meV	34	37	**30.8**	(31.3)
$\Delta\varepsilon$	meV	63	69	**58.0**	(58.9)
ZPVE	meV	1.7	**1.5**	**1.49**	(1.52)
μ	Debye	0.033	0.030	**0.029**	(0.029)
α	Bohr^3	0.235	0.092	**0.077**	(0.082)
$\langle R^2 \rangle$	Bohr^2	**0.073**	0.180	**0.072**	(0.075)
U_0	meV	14	19	**10.5**	(11.1)
U	meV	19	19	**10.6**	(11.2)
H	meV	14	17	**11.3**	(11.9)
G	meV	14	19	**12.2**	(12.7)
C_v	cal/molK	**0.033**	0.040	**0.032**	(0.033)

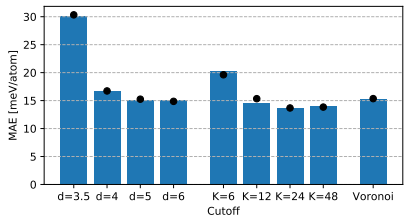


Figure 4: MAE of formation energy predictions on OQMD using different cutoffs (cutoff distance in Ångström, K-nearest and Voronoi) for constructing the graphs. The black dots show the mean absolute error for the validation set used for early stopping. The lowest error is 13.7 meV/atom using K-nearest cutoff with K=24.

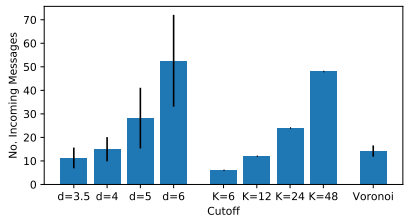


Figure 5: Average number of incoming messages per atom for all atoms in OQMD. The error bars show the standard deviation.

The prediction accuracy of the model when using different cutoff methods is shown in Figure 4. We also show the average number of incoming edges across all atoms in the OQMD dataset in Figure 5, which serves as a proxy for the computational complexity. The results shows that the K-nearest cutoff is more efficient in terms of achieving a low MAE for a given average number of edges per atom and we achieve the lowest error ($13.7\,\text{meV}$) with $K = 24$. This is not only caused by eliminating the "isolated atoms problem", because the Voronoi tessellation method is also without this problem. One reason could be that training the model is more stable when the number of incoming messages is constant.

6 Conclusion

We have proposed a novel neural message passing model for molecules and materials by extending the SchNet model with an edge update network that allows the information exchanged between atoms to be dependent on the sending and receiving atom. This simple extension leads to a remarkable improvement in prediction accuracy. By inspecting the learned filters for the molecular formation energy prediction task we found that the edge updates in the first layer has a higher dependence on the receiving atom than the sending atom. Through numerical simulations we demonstrate improvements in accuracy on formation energy prediction of molecules and materials across all three benchmark

datasets. We also highlight the problem of choosing the cutoff when constructing the graphs used as input for the model. We found that using K-nearest neighbors cutoff yields lower error than using a constant cutoff distance. We believe these results are important for applications of message passing neural networks for predicting properties of molecules and materials and hope to see more applications and further improvements of the model architecture in the future.

References

Battaglia, P. W., Pascanu, R., Lai, M., Rezende, D., and Kavukcuoglu, K. (2016). Interaction networks for learning about objects, relations and physics. 1612.00222.

Bergerhoff, G., Hundt, R., Sievers, R., and Brown, I. D. (1983). The inorganic crystal structure data base. *J. Chem. Inf. Comput. Sci.*, 23(2):66–69.

Bruna, J., Zaremba, W., Szlam, A., and LeCun, Y. (2013). Spectral networks and locally connected networks on graphs. 1312.6203.

Duvenaud, D., Maclaurin, D., Aguilera-Iparraguirre, J., Gómez-Bombarelli, R., Hirzel, T., Aspuru-Guzik, A., and Adams, R. P. (2015). Convolutional networks on graphs for learning molecular fingerprints. 1509.09292.

Faber, F. A., Hutchison, L., Huang, B., Gilmer, J., Schoenholz, S. S., Dahl, G. E., Vinyals, O., Kearnes, S., Riley, P. F., and von Lilienfeld, O. A. (2017). Prediction errors of molecular machine learning models lower than hybrid DFT error. *J. Chem. Theory Comput.*, 13(11):5255–5264.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. (2017). Neural message passing for quantum chemistry. In *International Conference on Machine Learning*, pages 1263–1272.

Gori, M., Monfardini, G., and Scarselli, F. (2005). A new model for learning in graph domains. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 2, pages 729–734 vol. 2.

Hansen, K., Biegler, F., Ramakrishnan, R., Pronobis, W., von Lilienfeld, O. A., Müller, K.-R., and Tkatchenko, A. (2015). Machine learning predictions of molecular properties: Accurate Many-Body potentials and nonlocality in chemical space. *J. Phys. Chem. Lett.*, 6(12):2326–2331.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition. 1512.03385.

Jain, A., Ong, S. P., Hautier, G., Chen, W., Richards, W. D., Dacek, S., Cholia, S., Gunter, D., Skinner, D., Ceder, G., and Persson, K. a. (2013). The Materials Project: A materials genome approach to accelerating materials innovation. *APL Materials*, 1(1):011002.

Kearnes, S., McCloskey, K., Berndl, M., Pande, V., and Riley, P. (2016). Molecular graph convolutions: moving beyond fingerprints. *J. Comput. Aided Mol. Des.*, 30(8):595–608.

Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization. 1412.6980.

Kirklin, S., Saal, J. E., Meredig, B., Thompson, A., Doak, J. W., Aykol, M., Rühl, S., and Wolverton, C. (2015). The open quantum materials database (OQMD): assessing the accuracy of DFT formation energies. *npj Computational Materials*, 1:15010.

Li, Y., Tarlow, D., Brockschmidt, M., and Zemel, R. (2015). Gated graph sequence neural networks. 1511.05493.

Ramakrishnan, R., Dral, P. O., Rupp, M., and von Lilienfeld, O. A. (2014). Quantum chemistry structures and properties of 134 kilo molecules. *Sci Data*, 1:140022.

Rogers, D. and Hahn, M. (2010). Extended-connectivity fingerprints. *J. Chem. Inf. Model.*, 50(5):742–754.

Ruddigkeit, L., van Deursen, R., Blum, L. C., and Reymond, J.-L. (2012). Enumeration of 166 billion organic small molecules in the chemical universe database GDB-17. *J. Chem. Inf. Model.*, 52(11):2864–2875.

Rupp, M., Tkatchenko, A., Müller, K.-R., and von Lilienfeld, O. A. (2012). Fast and accurate modeling of molecular atomization energies with machine learning. *Phys. Rev. Lett.*, 108(5):058301.

Rycroft, C. H. (2009). VORO++: a three-dimensional voronoi cell library in c++. *Chaos*, 19(4):041111.

Saal, J. E., Kirklin, S., Aykol, M., Meredig, B., and Wolverton, C. (2013). Materials design and discovery with High-Throughput density functional theory: The open quantum materials database (OQMD). *JOM*, 65(11):1501–1509.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. (2009). The graph neural network model. *IEEE Trans. Neural Netw.*, 20(1):61–80.

Schütt, K. T., Arbabzadah, F., Chmiela, S., Müller, K. R., and Tkatchenko, A. (2017a). Quantum-chemical insights from deep tensor neural networks. *Nat. Commun.*, 8:13890.

Schütt, K. T., Kindermans, P.-J., Sauceda, H. E., Chmiela, S., Tkatchenko, A., and Müller, K.-R. (2017b). SchNet: A continuous-filter convolutional neural network for modeling quantum interactions. 1706.08566.

Schütt, K. T., Sauceda, H. E., Kindermans, P.-J., Tkatchenko, A., and Müller, K.-R. (2017c). SchNet - a deep learning architecture for molecules and materials. 1712.06113.

Ward, L., Liu, R., Krishna, A., Hegde, V. I., Agrawal, A., Choudhary, A., and Wolverton, C. (2017). Including crystal structure attributes in machine learning models of formation energies via voronoi tessellations. *Phys. Rev. B Condens. Matter*, 96(2):024104.

Xie, T. and Grossman, J. C. (2018). Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties. *Phys. Rev. Lett.*, 120(14):145301.

APPENDIX B

Materials property prediction without atomic positions using graph neural networks

The following paper is ready for journal submission, but has not yet been submitted.

Materials property prediction without atomic positions using graph neural networks

Peter Bjørn Jørgensen, Estefanía Garijo del Río, Mikkel N. Schmidt, and Karsten Wedel Jacobsen

Computational materials screening studies require fast calculation of the properties of thousands of materials. The calculations are often performed with Density Functional Theory (DFT), but the necessary computer time sets limitations for the investigated material space. Therefore, the development of machine learning models for prediction of DFT calculated properties are currently of interest. A particular challenge for *new* materials is that the atomic positions are generally not known. We present a machine learning model for the prediction of DFT-calculated formation energies based on Voronoi quotient graphs and local symmetry classification without the need for detailed information about atomic positions. The model is implemented as a message passing neural network and tested on the Open Quantum Materials Database (OQMD) and the Materials Project database. The test mean absolute error is 20 meV on the OQMD database and 40 meV on Materials Project Database. The possibilities for prediction in a realistic computational screening setting is investigated on a dataset of 5976 $ABSe_3$ selenides with very limited overlap with the OQMD training set. Pretraining on OQMD and subsequent training on 100 selenides result in a mean absolute error below 0.1 eV for the formation energy of the selenides.

I. INTRODUCTION

Over the last decades, high-throughput computational screening studies have been employed to identify new materials within different areas as (photo-)electrochemistry [1–3], batteries [4, 5], catalysis [6, 7], and more [8–10]. Such studies are typically based on Density Functional Theory [11, 12] and because of computational requirements they are usually limited to some thousands or tens of thousands of materials. In order to investigate larger parts of the huge space of possible materials, new methods are needed to perform faster calculations or to guide the search in the material space in a more informed way.

One way to circumvent the computationally demanding DFT calculations is to use machine learning (ML) techniques to predict materials properties, and this approach has been explored intensively the last years. Several descriptors or fingerprints to describe the atomic structure of a material have been suggested including the partial radial distribution function [13] and the Coulomb matrix [14]. More involved fingerprints combining many atomic properties and crystal structure attributes based on Voronoi graphs have also been developed [15, 16], along with graph representations, which are directly mapped onto convolutional neural networks [17–19].

The use of ML to speed up DFT calculations may have several goals in a computational screening setting. If the atomic structure (i.e. the positions of all the atoms) of a material is known, ML may in principle provide the same information about the material as a DFT calculation would: structural stability, phonon dispersion relations, elastic constants etc. It might even in principle provide data of a better quality than standard (semi-)local DFT calculations, comparable to more advanced DFT calculations with hybrid functionals or even higher-level methods as recently demonstrated for molecules [20].

However, the atomic positions of *new* materials will generally not be known. If the atomic positions are known from experiment, the material is not really new (even though many of its properties might be unknown) and if the positions are obtained from a DFT calculations there is no need to use a ML prediction of already calculated properties.

Our focus here will be the prediction of properties of *new* materials where the detailed atomic positions are unknown, and since the most crucial property of a new material is its stability we shall concentrate on prediction of formation energies.

The obvious question of course then is, how we can describe or classify a crystalline material without knowing the explicit positions of the atoms. The most fundamental property of a material is its chemical composition, i.e. for a ternary material $A_xB_yC_z$, the identity of the elements A, B, and C and their relative appearance $x : y : z$. It turns out that based on this information alone a number of predictions about material stability can be made. Meredig et al. [21] demonstrated that it is possible to predict thermodynamic stability of new compounds with reasonable accuracy based on composition alone, and a number of new compound compositions were predicted and their structures subsequently determined. However, this approach of course has its limitations as it cannot distinguish between materials with the same composition but different crystal structures.

A rigorous classification of a crystalline material comes from its symmetry. Any periodic material belongs to one of the 230 space groups, and this puts restrictions on the possible atomic positions. In the simplest cases of, say, a unary material with one atom in the unit cell with space group Fm-3m (an fcc crystal), all atomic positions are determined up to a scaling of the volume. Similarly, the fractional positions (i.e. relative to the unit cell) of the atoms in materials with several elements can be determined entirely by symmetry as for example shown for $BaSnO_3$ in the cubic perovskite structure in Figure 1. More generally, scaled atomic positions may be fully or partially determined depending on their symmetry, and the symmetry properties can be expressed using the so-called Wyckoff sites. This classification was recently used by Jain and Bligaard [22] to build a machine learning

model based on only composition and the Wyckoff positions, i.e. without any detailed information about the atomic positions. They were able to achieve a mean absolute errors of about 0.07 eV/at on the prediction of the formation energy on a test dataset of more than 85000 materials.

Here, we shall develop a machine learning model, which does not require knowledge of the detailed atomic positions. However, unlike the model proposed by Jain and Bligaard, it will be based on local information about interatomic bonds and the symmetry of their environments. The bonds will be identified using Voronoi graphs and the symmetry will be classified using the Voronoi facets. The resulting model has a mean absolute error on the heats of formation for the OQMD database of only 21 meV and for the ICSD part of OQMD it is 38 meV.

In section II we describe the proposed graph representation based on quotient graphs and the classification of Voronoi facet point symmetry and in section III we investigate the relation between quotient graphs and prototypes based on data from OQMD. This is followed by an introduction of the machine learning model and the datasets in section IV and V respectively. The numerical results are presented in section VI and followed by the conclusions in section VII.

II. GRAPH REPRESENTATION

As representation for the machine learning algorithm we use the quotient graph as introduced by [23] and also used in [19]. The quotient graph is a finite graph representation of the infinite periodic network of atoms. Every atom in the unit cell corresponds to a vertex of the quotient graph. We denote the graph G and the set of N vertices $\{v_i\}_{i=1}^{N}$. When two atoms are connected in the network we draw an edge between the atoms in the quotient graph. In this work we use the Voronoi diagram to decide when two atoms are connected, specifically a pair of atoms are connected if they share a facet in the Voronoi diagram. Due to periodic boundary conditions a pair of atoms may share several facets and in this case there will be several edges between the atoms. When interatomic distances are available the edges are labeled with the distance between the atoms.

As an example we look at BaSnO3 in the perovskite structure as shown in Figure 1. This material has five atoms in the unit cell. After performing Voronoi tessellation we get a Voronoi cell for each atom in the unit cell as shown in Figure 2. The Voronoi diagram defines the edges in the quotient graph which is illustrated in Figure 3.

The Voronoi construction may result in the appearance of quite small facets. This is for example often the case for structures with high symmetry, where small displacements of the atoms introduce new facets. We remove these small facets and the corresponding connections in the graph by introducing a cutoff in the solid angle of the facet Ω_{cut}. We use $\Omega_{cut} = 0.2$, but as we shall see later the results are surprisingly stable with regards to increasing this value.

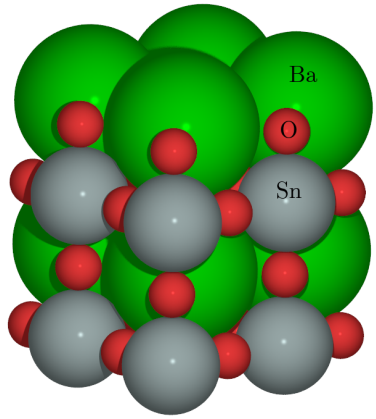


Figure 1. Structure of BaSnO3. The unit cell contains one Ba atom (green), one Sn atom (grey) and three O atoms (red).

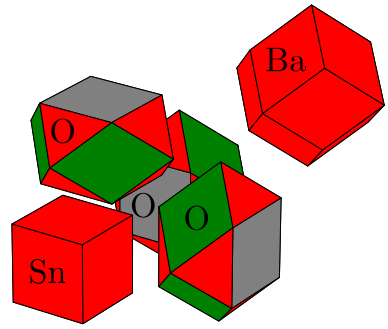


Figure 2. Voronoi cells of BaSnO3. The cells have been displaced for the visualization. The color of the facets corresponds to the atomic species of the neighbouring atom.

The graph is annotated with the symmetry group of each of the Voronoi facets. In the following section we describe this symmetry classification in more detail.

A. Symmetry Group Classification

To characterize the symmetry of an atomic environment we classify the symmetry of each Voronoi facet into the 9 non-trivial two-dimensional point groups $(C_2, C_3, C_4, C_6, D_1, D_2, D_3, D_4, D_6)$. The classification method is inspired by the symmetry measure introduced by Heijmans and Tuzikov [24]. Given the vertices of the two-dimensional Voronoi facet we go through the following procedure

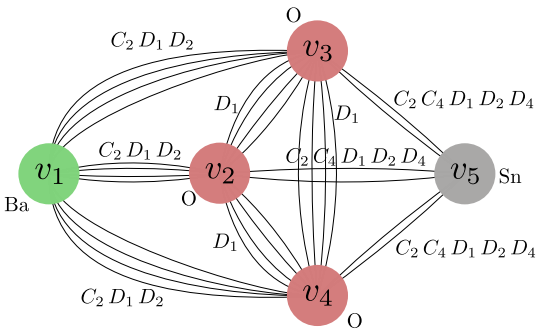


Figure 3. Quotient graph for BaSnO3. The edge labels shows the point groups of the corresponding facets of the Voronoi diagram. For this particular case, the repeated edges between vertices all have the same point groups, but in general they could be different.

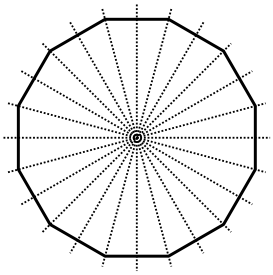


Figure 4. Mirror axes of a dodecagon.

1. Compute centroid and center the shape.

2. Search for mirror axis and align it with the x-axis if it exists.

3. For each point symmetry group apply all elements of the group and calculate the area of the convex hull of the new points generated by this procedure.

The symmetry measure is then the ratio between the area of the original shape and the area defined by the convex hull of the new vertices. When the symmetry measure for a given group is close to unity we label the facet as having this symmetry. See Figure 5 for an example shape and its symmetry measure for each group. The search for mirror axis in step 2 is done by computing the moment of inertia and test the two principal axes for mirror symmetry. When the moments of inertia are the same, for example when the shape is a regular polygon, the principal axes are arbitrary and we fall back to testing for mirror symmetry at all axes going through the centroid and either a vertex or a midpoint of a line segment. For a regular dodecagon these axes are illustrated in Figure 4.

III. GRAPH REPRESENTATION AND PROTOTYPES

In many applications prototypes are used as a descriptor for the overall structure of a material and as part of a computational screening procedure some of the atoms of the prototypes may be swapped with other elements. We want to assess whether there is a correspondence between the prototypes and Voronoi graphs, i.e. do two materials with the same prototype have the same Voronoi graph and do two materials with the same Voronoi graph have the same prototype? The question cannot be ultimately answered because prototype naming is not completely well-defined: in some cases several different prototypes are used to describe the same material, and many materials may not have prototypes attached to them. But we can show to which extent Voronoi graphs is aligned with the use of prototypes.

For this analysis we use the OQMD database and look at all unary, binary and ternary compounds that are labeled with a prototype. For each of these sets we want to look at the link between the graphs G and the prototypes P, i.e. if we know that a given structure has a specific prototype do we then also know which graph it has and vice versa. One way of measuring this is through the mutual information $I(G;P)$ of G and P. The mutual information is symmetric and can be computed as

$$I(G;P) = H(G) - H(G|P) \quad (1)$$
$$= H(P) - H(P|G), \quad (2)$$

where H denotes the entropy. The mutual information is thus the average decrease in entropy we get from knowing the other variable. We also compute the normalized mutual information known as the uncertainty coefficient $U(X|Y) = I(X;Y)/H(X)$ which can be seen as given Y what fraction of bits of X can we predict. The quantities for OQMD are shown for the unlabeled graph in Table I and for the graph labeled with rotation symmetries in Table II.

The uncertainty coefficient is close to 90% in most cases except for the Unary compounds $U(P|G)$. In this case structures with different prototypes map to the same graph and we may be discarding important structural information. Including symmetry information increases the number of unique graphs significantly, which implies that the uncertainty coefficient $U(G|P)$ decreases while $U(P|G)$ increases.

IV. NEURAL MESSAGE PASSING MODEL

In this section we introduce the machine learning model which takes the labeled graph as input and outputs an energy prediction as a scalar. We describe the model as a message passing framework on a graph, similarly to Gilmer *et al.* [25]. Denote the graph G with vertex features x_v and edge features ε_{vw} for an edge from

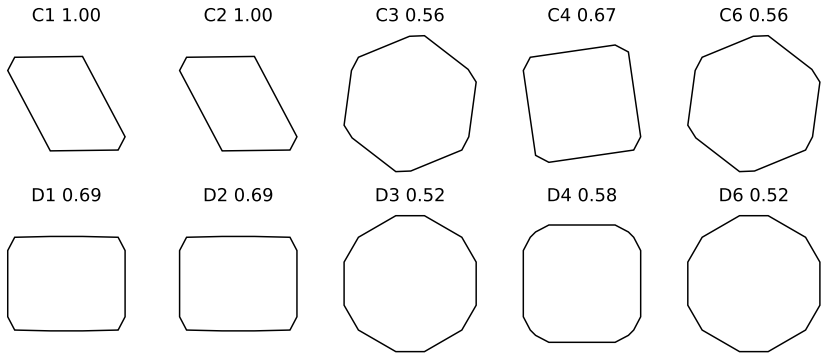


Figure 5. Convex hull of the shape in the top left corner after the symmetry operations of the corresponding groups has been applied. The label above each shape denotes the point group and the symmetry measure for that group.

	N	$\|G\|$	$\|P\|$	$H(G)$	$H(P)$	$I(G,P)$	$U(G\|P)$	$U(P\|G)$
Unary	1487	85	67	4.4	4.7	3.7	0.84	0.80
Unary ICSD	196	46	49	4.2	4.2	3.8	0.90	0.90
Binary	53528	1318	871	4.3	4.5	3.8	0.90	0.86
Binary ICSD	5862	1219	850	8.2	8.0	7.6	0.92	0.95
Ternary	339960	4006	1754	2.0	1.9	1.8	0.91	0.98
Ternary ICSD	11500	3487	1740	10.0	9.1	8.8	0.88	0.97

Table I. Correspondence between Voronoi graphs and prototypes in OQMD without symmetry labels. N denotes the number of entries, $|G|$ the number of unique Voronoi graphs and $|P|$ the number of different prototypes.

vertex v to vertex w. Each vertex has a hidden state h_v^t at "time" (or layer) t and we denote the edge hidden state e_{vw}^t. The hidden states of vertices and edges are updated in a number of interaction steps T. In each step the hidden state of vertices are updated in parallel by receiving and aggregating messages from neighbouring vertices. The messages are computed by the message function $M_t(\cdot)$ and the vertex state is updated by a state transition function $S_t(\cdot)$, i.e.

$$m_v^{t+1} = \sum_{w \in N(v)} M_t\left(h_v^t, h_w^t, e_{vw}^t\right), \tag{3}$$

$$h_v^{t+1} = S_t\left(h_v^t, m_v^{t+1}\right), \tag{4}$$

where $N(v)$ denotes the neighborhood of v, i.e. the vertices in the graph that has an edge to v. The edge hidden states are also updated by an edge update function $E_t(\cdot)$ that depends on the previous edge state as well as the vertices that the edge connects:

$$e_{vw}^{t+1} = E_t(h_v^t, h_w^t, e_{vw}^t). \tag{5}$$

After T interaction steps the vertex hidden state represents the atom type and its chemical environment. We then apply a readout function $R(\cdot)$ which maps the set of vertex states to a single entity

$$\hat{y} = R\left(\left\{h_v^T \in G\right\}\right) \tag{6}$$

The readout function operates on the set of vertices and must be invariant to the ordering of the set. This is often achieved simply by summing over the vertices. In some architectures the final edge states are also included as an argument to the readout function. The message function $M_t(\cdot)$, state transition function $S_t(\cdot)$, edge update function $E_t(\cdot)$ and readout function $R(\cdot)$ are implemented as neural networks with trainable weight matrices. To fully define the model we just need to define these functions and a number of models can be cast into this framework. We use different weight matrices for each time step t, however in some architectures the weights are shared between layers to reduce the number of parameters.

In this work we use the model proposed in our prior work [26]. The model can be seen as an extension of the SchNet model [18], because the only difference is the inclusion of an edge update network. The message function is only a function of the sending vertex and can be written as

$$M_t(h_w^t, e_{vw}^t) = (W_1^t h_w^t) \odot g(W_3^t g(W_2^t e_{vw}^t)), \tag{7}$$

where $\odot$ is element-wise multiplication and $g(x)$ is the activation function, more specifically the shifted soft-plus function $g(x) = \ln(\mathrm{e}^x + 1) - \ln(2)$. It can be seen as a smooth version of the more popular rectified linear unit. As an edge update network we use a two layer neural

	N	$\lvert G\rvert$	$\lvert P\rvert$	$H(G)$	$H(P)$	$I(G,P)$	$U(G\lvert P)$	$U(P\lvert G)$
Unary	1487	316	67	6.6	4.7	4.4	0.67	0.94
Unary ICSD	196	90	49	5.4	4.7	4.2	0.77	0.99
Binary	53528	2491	871	5.6	4.5	4.3	0.77	0.96
Binary ICSD	5862	1997	850	9.1	8.0	7.8	0.86	0.98
Ternary	339960	6927	1754	2.1	1.9	1.8	0.86	0.99
Ternary ICSD	11500	5169	1740	10.9	9.1	10.0	0.82	0.99

Table II. Correspondence between Voronoi graphs and prototypes in OQMD with symmetry labels. N denotes the number of entries, $|G|$ the number of unique Voronoi graphs and $|P|$ the number of different prototypes.

network and the input is a concatenation of the sending and receiving vertex states and the current edge state.

$$e_{vw}^{t+1} = E_t(h_v^t, h_w^t, e_{vw}^t) = g(W_{\mathrm{E2}}^t g(W_{\mathrm{E1}}^t(h_v^t; h_w^t; e_{vw}^t))), \tag{8}$$

where $(\cdot;\cdot)$ denotes vector concatenation. This choice of edge update network means that the edge state for each of the two different directions between a pair of vertices become different after the first update. The state transition function is also a two layer neural network. It is applied to the sum of incoming messages and the result is added to the current hidden state as in Residual Networks [27]:

$$S_t\left(h_v^t, m_v^{t+1}\right) = h_v^t + W_5^t g(W_4^t m_v^{t+1}), \tag{9}$$

After a number of interaction steps T we apply a readout function for which we use a two layer neural network that maps the vertex hidden representation to a scalar and finally we average over the contribution from each atom, i.e.

$$R\left(\left\{h_v^T \in G\right\}\right) = \frac{1}{N}\sum_{h_v^T \in G} W_7 g(W_6 h_v^T), \tag{10}$$

In other words an atom and its chemical environment is mapped to an energy contribution.

A. Initial Vertex and Edge Representation

The initial vertex hidden state h_v^0 depends on the atomic number of the corresponding atom. The atomic number is used to look up a vector representation for that atom. Using a hidden representation of size 256 the initial hidden state is thus the result of a lookup function $\ell(x) : \mathbb{N} \to \mathbb{R}^{256}$. The weights in the vector representation are also trained during the optimization.

We use the model on three different levels of available information. In the most ignorant scenario we have no labels on the edges of the graph and in this case the edge update function (8) just ignores the edge representation on the first layer, i.e. e_{vw}^0 is a "vector" of length 0 and $e_{vw}^t, t \in 1, \ldots, T$ are vectors of length 256. The next level of information is to include the point group information as described in section II A. There are 9 non-trivial point groups and we encode this information as an indicator vector of length 9, where 1 means that the corresponding facet belongs to the given point group. Finally we also run numerical experiments with the full spatial information for which the edges of the quotient graph are labeled with the interatomic distance. The distances are encoded by expanding them in a series of exponentiated quadratic functions as also done in [17, 18, 26]:

$$(e_{vw}^0)_k = \exp\left(-\frac{(d_{vw} - (-\mu_{\min} + k\Delta))^2}{2\Delta^2}\right), k = 0 \ldots k_{\max} \tag{11}$$

where $\mu_{\min}$, Δ, and $k_{\max}$ are chosen such that the centers of the functions covers the range of the input features. This can be seen as a soft 1-hot-encoding of the distances, which makes it easier for a neural network to learn a function where the input distance is uncorrelated with the output. In the experiments we use $\mu_{\min} = 0$, $\Delta = 0.1$, and $k_{\max} = 150$.

V. DATASETS

For the numerical experiments we use two publicly available datasets and one dataset we generate.

a. The Materials Project [28] This dataset contains geometries and formation energies of 86 680 inorganic compounds with input structures primarily taken from the The Inorganic Crystal Structure Database (ICSD) [29]. We use the latest version of the database (version 2018.11). The number of examples is reduced to 86 579 after we exclude all materials with noble gases (He, Ne, Ar, Kr, Xe) because they occur so infrequently in the dataset that we consider them as outliers. This brings the number of different elements in the dataset down to 84.

b. Open Quantum Materials Database (OQMD) [30, 31] Is also a database of inorganic structures and we use the currently latest version (OQMD v1.2) available on the project's website. Again we consider materials with noble gases as outliers and we also exclude highly unstable materials with a heat of formation of more than 5 eV/atom. Some entries in the database are marked as duplicates and we filter them in the following way: when a set of duplicates is encountered we use the first entry of the database, but if the standard deviation of

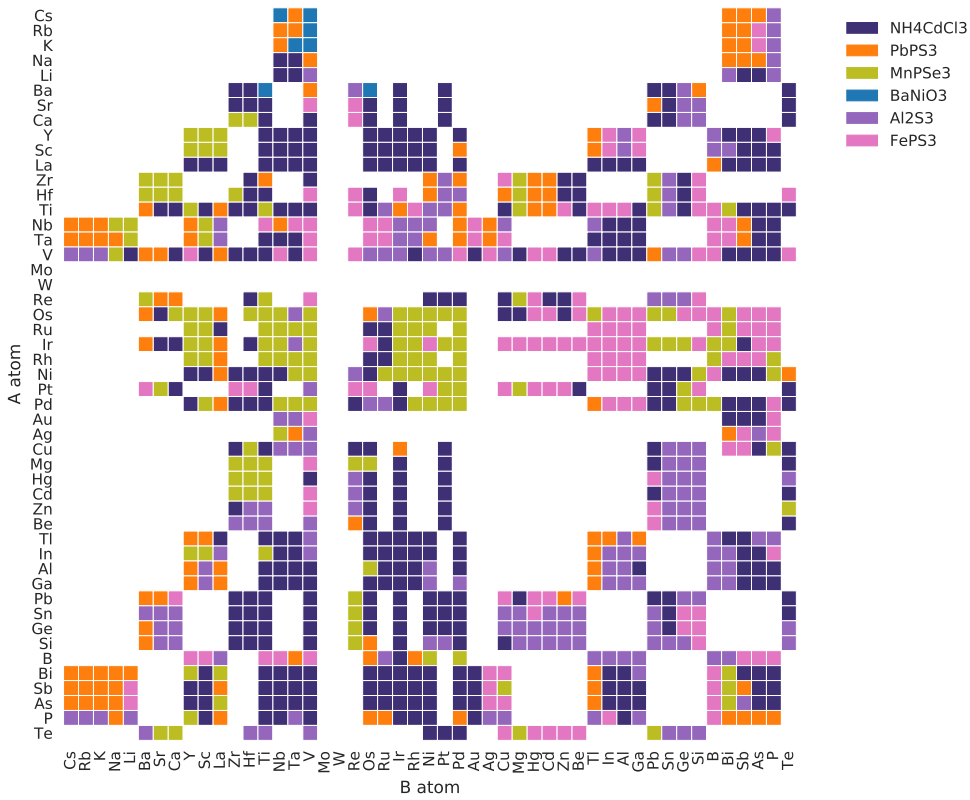


Figure 6. Map of the most stable prototype for each composition $ABSe_3$. The compositions that do not fulfill the valence rule have not been studied and thus, they are not colored.

the calculated heat of formation exceeds 0.05 eV/atom we discard the whole set of duplicates. This leaves us with a total of 562 134 entries.

For both datasets we split the entries into five splits of equal size to be used for 5-fold cross-validation, where the machine is trained on 4/5 of the data, and the remaining 1/5 is used for testing. For OQMD we also distribute the entries of OQMD that originates from ICSD equally between the five folds.

c. Ternary Selenides $ABSe_3$ For further testing, we have developed a third dataset of selenides. The intention behind this set is to test the ability of the model in a realistic computational screening setting. This dataset has only very limited overlap with OQMD, and predictions are made exclusively based on the symmetry labeled graphs of the new materials without any detailed information about the atomic coordinates.

The dataset contains the structures and formation energies of 5976 ternary selenides with stoichiometries $ABSe_3$, where A and B are different transition metals in six different prototypes.

The procedure for generating this dataset resembles the one presented in reference [3]. We start by looking up the $ABSe_3$ compounds reported in ICSD [29], and selecting the 6 prototypes that appear more than once: hexagonal $P6_3/mmc$ structure of $BaNiO_3$, orthorhombic $Pnma$ structure of NH_4CdCl_3/Sn_2S_3, monoclinic $C2/m$ $FePS_3$, monoclinic Pc structure of $PbPS_3$, trigonal $R\bar{3}$ structure of $MnPSe_3$ and hexagonal $P6_1$ structure of Al_2S_3.

These structures are then used as templates, and we substitute the transition metal atoms A and B by 49 transition metals. Here, we avoid for simplicity Cr, Mn, Fe and Co, which usually lead to structures with large magnetic moments. We also limit ourselves to those combinations $ABSe_3$ for which the valences of cations and anions add up to zero. This leads to a total of 512 $ABSe_3$ compounds: 484 ternaries, which are then studied in 12 structures (6 for the $ABSe_3$ and 6 for the $BASe_3$) and 28 binaries, for which we study 6 different structures. A map to the compositions and structures studied can be

found in figure 6.

The resulting 5976 structures have then been relaxed using Density Functional Theory (DFT) as implemented in the codes ASE [32] and GPAW [33]. We perform two different kinds of electronic structure calculations: a coarse-grained calculation with the exchange-correlation functional PBEsol [34] for the steps of the optimization and a fined-grained at the relaxed structure with the PBE exchange-correlation functional [35]. The cuoff energy for the plane wave basis set used to expand the wave functions is 800 eV in both cases. For the sampling of the Brillouin zone we use a Monkhorst–Pack mesh [36] with a density of $5.0/(\text{Å}^{-1})$ k-points in each direction for the relaxation steps and of $8.0/(\text{Å}^{-1})$ k-points for the refined calculation at the relaxed structure. All structures have been relaxed until the forces on the atoms have reached at least 0.05 eV/Å.

VI. NUMERICAL RESULTS AND DISCUSSION

To assess the loss in accuracy going from full spatial information to unlabeled quotient graph we train/test the model in three different settings as already mentioned in section IV A. In the most ignorant setting the quotient graph has only unlabeled edges. On the next level we label the edges with the symmetry of the corresponding Voronoi facet. With full spatial information the edges of the quotient graph are labeled with the distance between the atoms. The model is trained with the Adam optimizer [37] for up to 10×10^6 steps using a batch size of 32. The initial learning rate is 1×10^{-4} and it is decreased exponentially so at step s the learning rate is $10^{-4} \cdot 0.96^{\frac{s}{10^5}}$. When training on OQMD and Materials Project we use 5000 examples from the training data for early stopping. More specifically this validation set is evaluated every 50 000 steps and if the mean absolute error (MAE) has not improved for 1×10^6 steps the training is terminated. When training on the ternary selenides $ABSe_3$ dataset the 10% of the training data is used as a validation set and the validation set is evaluated every training epoch. In some of the results we use a model that has been pretrained on OQMD. In that case the model is trained on 4 out of 5 OQMD folds until the stopping criterion is met and the weights of the model are then used as initialization for training on the selenides dataset.

A. OQMD

The mean absolute errors (MAE) and root mean squared errors (RMSE) of the test set predictions are shown in Table III and the MAE is further visualized in Figure 7. As expected, the lowest prediction errors are obtained with the model where distance information is provided. If we focus on the OQMD the overall MAE is as low as 14 meV with distance information. This is lower than the SchNet-model [18] by almost a factor of two because of the edge updates as discussed in Ref. [26]. Two versions of the models without distance information are also shown. In one of them the symmetry information has not been used, but in the other one the symmetry classification of the Voronoi facets has been included as edge information. These two models do of course have larger errors than the one benefiting from the distance information, but still the error is surprisingly small. The MAE is only 20 meV for the model using symmetry information. For comparison the results for the model proposed by Ward et al. [15] is also shown in the figures (labeled V-RF for Voronoi - random forest). This model also builds on a Voronoi graph construction, but since the fractional areas of the Voronoi cells are provided, information about the distances are provided. Furthermore, many other attributes are added as information to the random forest algorithm applied. When this machine is applied to OQMD (using the same 5-fold splitting of the data as applied to the other algorithms) the resulting error is considerably larger, 85 meV, for all of OQMD.

To understand more about the behavior of the ML algorithms investigated here, we have considered the test

	Dist.	Sym	No sym	V-RF
OQMD all	14	20	26	85
OQMD unary	58	108	128	85
OQMD binary	30	39	60	86
OQMD ternary	14	19	23	80
ICSD all	24	38	45	113
ICSD unary	56	75	119	72
ICSD binary	32	47	58	118
ICSD ternary	22	34	39	109
Matproj all	26	40	43	84
Matproj unary	96	144	179	127
Matproj binary	48	69	73	99
Matproj ternary	27	40	43	87

Table III. MAE of test set energy predictions in meV/atom. The ICSD results are for the model trained on OQMD and tested only on the ICSD part of OQMD.

	Dist.	Sym	No sym	V-RF
OQMD all	54	70	80	173
OQMD unary	184	257	342	190
OQMD binary	89	98	138	162
OQMD ternary	52	68	71	131
ICSD all	81	98	111	188
ICSD unary	262	222	353	180
ICSD binary	73	111	129	202
ICSD ternary	88	85	102	182
Matproj all	72	115	122	172
Matproj unary	246	349	467	289
Matproj binary	120	180	192	203
Matproj ternary	65	108	111	181

Table IV. RMSE of test set energy predictions in meV/atom. The ICSD results are for the model trained on OQMD and tested only on the ICSD part of OQMD.

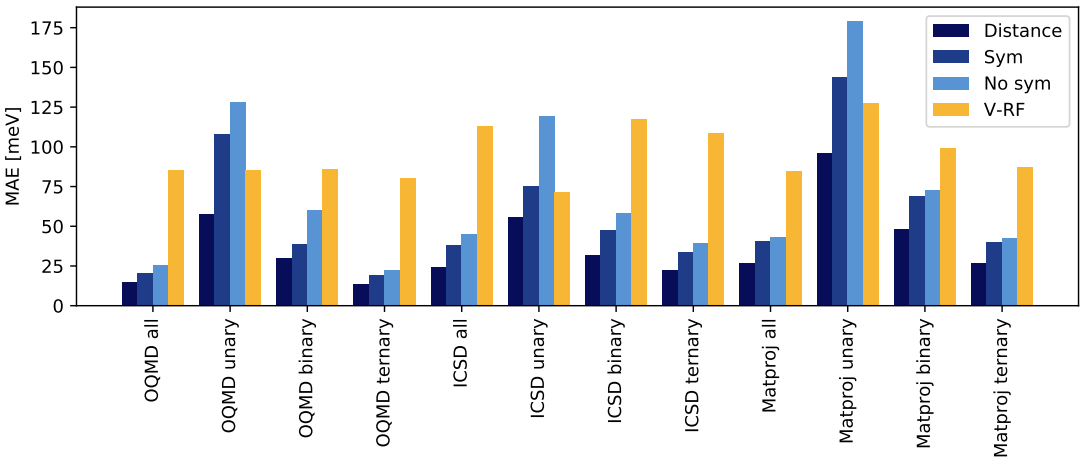


Figure 7. The figure shows the data in Table III

errors on different subsets of OQMD and also on the Material Project database [28]. Let us first note that the OQMD contains two different types of structure sources. One type, which gives rise to the largest number of materials, consists of a number of fixed crystal structures or prototypes decorated by the different chemical elements. There are 16 elemental prototypes, 12 binary ones, and 3 ternary ones. For two of the ternary ones, one of the elements is predefined to be oxygen. This generates a very large number of materials of varying composition and stability, but in a fairly small number of different crystal structures. The other type of structures comes from materials from the experimental ICSD database. This group is characterized by a much greater variation in the crystal structures, but is naturally limited to mostly stable materials, since they have been experimentally synthesized.

We first consider the test error on the subsets of OQMD consisting of the unary, binary, and ternary systems, and we shall focus on the model where the symmetry information is included, but the distances are not. As can be seen from Table III, the test error is considerably larger on the unary systems (108 meV) than on the database as a whole. This also holds for the binary ones but to a smaller degree (39 meV). It is not clear to us at the moment exactly why this is so, but we shall discuss some possible explanations. The unary and binary systems only constitute a fairly small part of the total database, and the weight of these systems during the training is therefore also limited. Another factor may be that a large fraction of the unary and binary systems belong to the group of materials where the crystal structures are systematically generated as mentioned above. This means that many rather "artificial" and unstable materials are generated, where the atoms are situated in environments, which do not occur in reality, and the resulting energies may be far above more stable structures. This could potentially be difficult for the machine to learn.

B. ICSD/OQMD

Table III also shows the results for the ICSD subset of the OQMD database. The results shown are for the model trained on all of OQMD but tested only on the ICSD subset. The overall MAE is seen to be roughly a factor of two larger than for all of OQMD. This is probably due to the fact that the ICSD is a subset with a large variation of structures and this makes prediction more difficult on average. We see the same trend as for all of OQMD, that the error decreases going from unary to binary to ternary systems. For the unary systems the test error is in fact lower for the ICSD subset than for all of OQMD, which may be due to the fact that physically artificial high-energy systems appear in OQMD but not in ICSD. For the binary systems there is a balance: the ICSD does not contain so many high-energy systems, which could make predictions better, but on the other hand the larger variation of crystal structures is more difficult to predict.

C. Materials Project Database

The models have also been trained and tested on the Materials Project dataset [28]. The overall error is fairly similar to the one obtained for the ICSD subset of OQMD as might be expected since the Materials Project is also based on mostly materials from the ICSD. The errors for the unary and binary subsets are somewhat larger for the

Materials Project database. This might be due to the fact that the machine trained on OQMD benefits from the larger number of systematically generated unary and binary systems in that database.

D. RMSE vs. MAE

The root-mean-square-errors are shown in Table IV. In all cases the values are considerable higher than the MAE. This is an indication that the distributions are non-Gaussian, and as we shall see in the following examples that a significant number of outliers exist. The outliers might be due to limitations of the model but could also appear because of problematic entries in the database as also discussed by Ward et al. [15].

E. Solid angle cutoff of Voronoi facets

The above results are all calculated using a cutoff of the Voronoi facet solid angle of $\Omega_{cut} = 0.2$. However, the results are almost independent of the value as shown in Figure 8, where the MAE on all of OQMD is shown for the model where symmetry but no distance information is included. We see that the error decreases slightly when small facets are removed with $\Omega_{cut} = 0.2$, and increases only slowly when Ω_{cut} is further increased. We take this as an indication that the connectivity of the material is well described even when the graph is reduced to essentially include only nearest neighbor bonds.

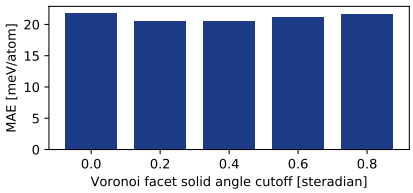


Figure 8. Prediction error vs Voronoi facet solid angle cutoff Ω_{cut} for the model using symmetry labels.

F. ABO_3 materials in OQMD

We now consider the subset of all oxides in the OQMD with the composition ABO_3. We shall investigate to which extent the model is able to predict the right ground state structure for a given composition. We first show the overall prediction for the 12935 materials of this type in OQMD in Figure 9. We again use the model with symmetry-labeled graphs without distance information. The MAE is 35 meV, which is about the same value as the one for the subset of ternaries in ICSD (34 meV). The

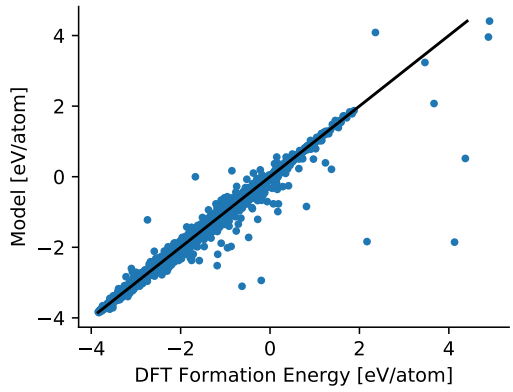


Figure 9. Predictions on 12395 ABO_3 structures of OQMD (MAE=35, RMSE=110 meV/atom)

RMSE is again significantly higher (110 meV) because of severe outliers as can be seen in the plot.

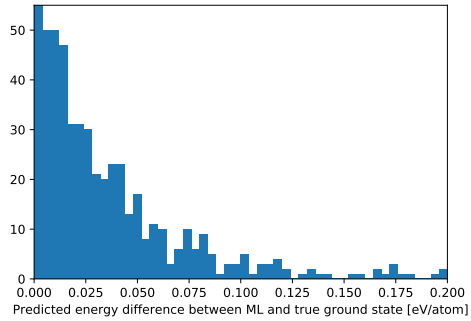
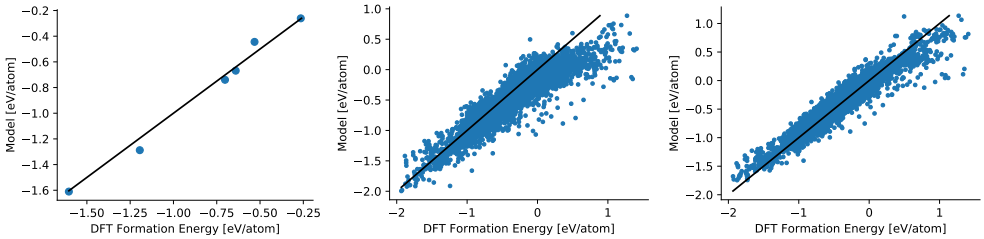


Figure 10. Predicted energy difference between the DFT ground state and the ML ground state: $\Delta E = E^{ML}(G_{\mathrm{DFT}}) - E^{ML}(G_{\mathrm{ML}})$ for the ABO_3 materials in OQMD. The total number of compositions is 2646. The peak at zero is much higher than shown in the graph. It corresponds to the 2113 compositions, where the right ground state is predicted. For the remaining 533 compositions the mean absolute difference is 42 meV/atom.

We now ask the following question: given a composition (A,B) the model predicts a ground state structure G_{ML}. If we are going to investigate this structure and other low energy structures with DFT, how high up in energy (as predicted by the model) do we have to go before we find the DFT ground state structure G_{DFT}? We only include entries for which there is more than one structure (12329/12395) and the average number of structures per composition is 4.7 The energy difference $\Delta E = E^{ML}(G_{\mathrm{DFT}}) - E^{ML}(G_{\mathrm{ML}})$ of course varies from



(a) Predicted energy for selenides in OQMD (MAE=43 RMSE=56 meV/atom)

(b) Predicted energy for selenides (initial structures) using model trained on OQMD (MAE=165 RMSE=222 meV/atom)

(c) Predicted energy for selenides (initial structures) using model pretrained on OQMD and then on 100 selenides (MAE=96 RMSE=133 meV/atom)

Figure 11. Model predictions on $ABSe_3$ structures.

system to system, and the distribution is shown in Figure 10. The mean absolute difference (MAD) of this distribution is very small, only 8 meV, and the maximum error is a clear outlier at 0.98 eV. The reason for the small MAD is that for 2113 out of the 2646 compositions the correct ground state is predicted, however, in many cases because only two structures exist in the database for a given composition. If we only look at the 533 compositions for which the ML model predicts the wrong ground state the MAD is 42 meV/atom. For comparison the energy prediction for the ground state structures has an MAE of 28 meV/atom. The low MAD value of 42 meV is promising for applications to computational screening. It sets an energy scale for how many structures have to be investigated by DFT to identify the DFT ground state after the model predictions have been generated.

G. $ABSe_3$ selenides

The last dataset we shall consider consists of selenides with the $ABSe_3$ composition as discussed in the section about the datasets. This dataset is considerably more challenging for two reasons. Firstly, there is very little overlap between this dataset and the training dataset OQMD. Only 6 materials are shared between the two datasets, and the test predictions for these are shown in Figure 11a. The MAE is 43 meV, and the RMSE is also low, only 56 meV. The second challenge is, that we shall now use the model to make predictions based on the initial graph before relaxations. The 6 different prototypes in the dataset each have a graph in the original material giving rise to the naming of the prototype. For example, one of the types is hexagonal $P6_3/mmc$ structure of $BaNiO_3$, so for predictions in this structure we shall use the graph of $BaNiO_3$. Some of the prototype structures have a fair number of atoms in the unit cell (up to 20) and a low symmetry (monoclinic), which means that there are many free atomic coordinates that are optimized during relaxation. This leads to frequent modifications of the graph during relaxation.

Figure 11b shows the model predictions based on the initial prototype graphs versus the DFT energies of the resulting optimized structures. The MAE is 165 meV, which is considerably higher than the value for the oxides. Particularly large deviations are seen for large and positive heats of formation. In a computational screening setting this might not be an issue because the high-energy materials are going to be excluded anyway. The RMSE is only a factor $222/165 = 1.35$ larger than the MAE, which is due to the small number of outliers compared to for example the oxides (Figure 9).

The prediction quality can be significantly improved by additional training on the selenide dataset. Even a limited number of data points have a considerable effect. This is to be expected since the overlap between the selenide dataset and the OQMD is only 6 materials as mentioned above. Figure 11c shows the model-DFT comparison if the model is first trained on the OQMD dataset and then subsequently trained on 100 materials out of the 5976 selenides in the database. The MAE is reduced from 165 meV to 96 meV bringing the error down to a value comparable to the error between DFT and experiment [31].

The effect of additional training on the selenide dataset is shown as a function of training set size on a logarithmic scale in Figure 12. The points on the y-axis correspond to the situation without any additional training. As can be seen, a small amount of additional training leads to significant reduction of the prediction error. The red curve corresponds to the situation discussed above where the model is first trained on OQMD, and then further trained on the initial graphs (but relaxed energies) for part of the selenides. For comparison, the green curve shows the prediction error, when the training and prediction is based on the final graph. Using the initial graphs instead of the final graphs gives rise to only a slightly higher MAE. This is encouraging for the potential use of the approach in computational screening studies, where predictions have to be based on the initial prototype structures to avoid

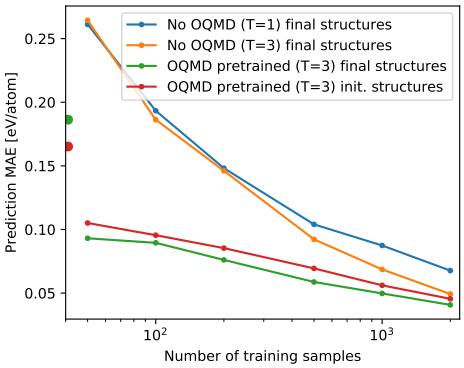


Figure 12. Predictions on $ABSe_3$ structures with increasing number of training samples. The unconnected points correspond to the model only trained on OQMD final structures, i.e. the pretrained model.

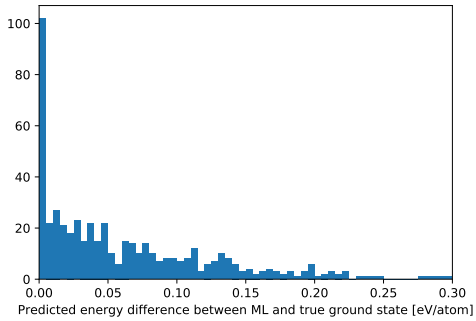


Figure 13. Predicted energy difference between the ML ground state and the true ground state, $\Delta E = E^{ML}(G_{\mathrm{DFT}}) - E^{ML}(G_{\mathrm{ML}})$, for the selenide dataset. The mean absolute difference is 62 meV/atom.

the computationally costly DFT calculations.

As a baseline we also show the results of the model if it is trained exclusively on the selenide dataset (orange curve). As expected the MAE is much larger than for the pretrained model for small amounts of data. For larger training sets the MAE drops gradually and with a dataset size of about 500 materials the prediction error is comparable to the one for the OQMD-pretrained model, which is trained on an additional 50 selenides. We ascribe the rather successful performance of the model without pretraining at large training set sizes to the systematic character of the dataset: only 6 different crystal structures are represented and they are systematically decorated with a particular subset of atoms. The last model (blue curve) is again only trained on the selenide dataset, but now only one interaction step ($T = 1$) is performed in the message passing neural network in contrast to the three iterations used otherwise. The performance is seen to be rather similar to the model with $T = 3$ up to a training dataset size of 300. With only one iteration in the network information about the identity of neighboring atoms is exchanged, and this is apparently sufficient to roughly characterize the 6 crystal structures. At larger training set sizes, where the prediction error is smaller, the network with three iterations outperforms the one with only one iteration.

Figure 13 shows the distribution of the predicted energy difference between the DFT ground state structure and the ML predicted ground state structure, $\Delta E = E^{ML}(G_{\mathrm{DFT}}) - E^{ML}(G_{\mathrm{ML}})$ for the selenide dataset. Only in 79 out of the 512 compositions, the model predicts the DFT ground state. This in not particularly impressive, since random prediction of a structure would give roughly $512/12 \approx 43$ correct predictions. However, the dataset have many low-lying energy structures, where even full DFT calculations cannot be expected to necessarily predict the correct ground state structure. This was investigated in more detail in a similarly generated dataset of ABS_3-sulfides used for computational screening of water-splitting materials [3]. The mean absolute difference is only 62 meV/atom with a maximum error of 0.3 eV/atom. The low mean value is clearly promising for future applications to computational materials screening.

VII. CONCLUSIONS

In summary, we have proposed a ML model for the prediction of the formation energy of crystalline materials based on Voronoi quotient graphs and a local symmetry description. It uses a message passing neural network with edge updates. The model is independent of the detailed atomic positions and can therefore be used to predict the formation energy of new materials, where the detailed structure is unknown.

The model test MAE is very small (20 meV) on the OQMD dataset, and a factor of two larger (38 meV) on the ICSD subset of OQMD. To test the model in a realistic materials screening setting, we created a dataset of 6000 selenides with very small overlap with the OQMD. The model pretrained on OQMD and applied to the selenides shows an MAE of 165 meV. This value can be lowered to 96 meV with an additional training on 100 selenides. Further training can lower the MAE to below 50 meV.

Based on the results we conclude, that is possible to develop ML models with position-independent descriptors, which are useful for realistic materials screening studies. However, extrapolation from OQMD to other datasets is challenging. One reason for this may be, as pointed out before, that the OQMD is composed of materials of two types: Some are generated systematically in rather few predefined crystal structures while others

come from ICSD. (There is of course a significant overlap between the two types). The first type is characterized by a large variation in stability, but low variation in crystal structures, while the second type is the opposite: the experimentally observed materials in ICSD exhibit a large variation in crystal structures, but they are all (except for some high-pressure entries) stable low-energy materials. This bias might limit the extrapolation to datasets which contain structures weakly represented in OQMD and with element combinations, which are far from stable. One way forward could be to create datasets with less bias so that unstable materials are represented in a greater variety of structures.

We see a number of potential improvements of the proposed model. More symmetry information could be included using for example Wyckoff positions [22] or additional graph edges describing symmetry relations. Furthermore, it is possible to label the quotient graphs with crystal translation information so that the infinite graph can be reconstructed [38]. This would make the crystal description more unique.

Perhaps the model could also learn the atomic positions from the graph representation. The latest developments in generative models have succeeded in generating small molecules in 3D space [39]. By combining this kind of model with the restrictions imposed by the connectivity and symmetries described by the quotient graph (see for example [40, 41]) it might be possible to directly predict the atomic positions without running DFT relaxations.

Another useful extension would be to model uncertainties in the prediction. Even though the datasets used here have a relatively high number of entries they only contain a tiny fraction of the chemical space. If the model could learn what it does not know it would be very useful in an active learning setting where DFT calculations could be launched by the model to explore areas of the chemical space with high uncertainty. A promising direction for uncertainty modeling is to use ensembles of neural networks where different techniques can be considered to ensure diversity between ensemble members [42–45].

VIII. ACKNOWLEDGMENTS

We would like to thank Peter Mahler Larsen for helpful discussions. We acknowledge support from the VILLUM Center for Science of Sustainable Fuels and Chemicals which is funded by the VILLUM Fonden research grant (9455) and thanks to Nvidia for the donation of one Titan X GPU.

[1] I. E. Castelli, T. Olsen, S. Datta, D. D. Landis, S. Dahl, K. S. Thygesen, and K. W. Jacobsen, Energy Environ. Sci. **5**, 5814 (2012).
[2] Y. Wu, P. Lazic, G. Hautier, K. Persson, and G. Ceder, Energy & Environmental Science **6**, 157 (2012).
[3] K. Kuhar, A. Crovetto, M. Pandey, K. S. Thygesen, B. Seger, P. C. K. Vesborg, O. Hansen, I. Chorkendorff, and K. W. Jacobsen, Energy Environ. Sci. **10**, 2579 (2017).
[4] A. Urban, D.-H. Seo, and G. Ceder, npj Computational Materials **2**, 16002 (2016).
[5] M. Aykol, S. Kim, V. Hegde, D. Snydacker, Z. Lu, S. Hao, S. Kirklin, D. Morgan, and C. Wolverton, Nature Communications **7** (2016), 10.1038/ncomms13779, cited By 38.
[6] M. Andersson, T. Bligaard, A. Kustov, K. Larsen, J. Greeley, T. Johannessen, C. Christensen, and J. K. Nørskov, Journal of Catalysis **239**, 501 (2006).
[7] J. K. Nørskov, T. Bligaard, J. Rossmeisl, and C. H. Christensen, Nature Chemistry **1**, 37 (2009).
[8] N. Mounet, M. Gibertini, P. Schwaller, D. Campi, A. Merkys, A. Marrazzo, T. Sohier, I. E. Castelli, A. Cepellotti, G. Pizzi, and N. Marzari, Nature Nanotechnology **13**, 246 (2018).
[9] S. Curtarolo, G. L. W. Hart, M. B. Nardelli, N. Mingo, S. Sanvito, and O. Levy, Nature Materials **12**, 191 (2013).
[10] K. Alberi, M. B. Nardelli, A. Zakutayev, L. Mitas, S. Curtarolo, A. Jain, M. Fornari, N. Marzari, I. Takeuchi, M. L. Green, M. Kanatzidis, M. F. Toney, S. Butenko, B. Meredig, S. Lany, U. Kattner, A. Davydov, E. S. Toberer, V. Stevanovic, A. Walsh, N.-G. Park, A. Aspuru-Guzik, D. P. Tabor, J. Nelson, J. Murphy, A. Setlur, J. Gregoire, H. Li, R. Xiao, A. Ludwig, L. W. Martin, A. M. Rappe, S.-H. Wei, and J. Perkins, Journal of Physics D: Applied Physics **52**, 013001 (2018).
[11] P. Hohenberg and W. Kohn, Physical Review **136**, 864 (1964).
[12] W. Kohn and L. J. Sham, Physical Review **140**, 1133 (1965).
[13] K. T. Schütt, H. Glawe, F. Brockherde, A. Sanna, K. R. Müller, and E. K. U. Gross, Physical Review B **89**, 205118 (2014).
[14] F. Faber, A. Lindmaa, O. A. von Lilienfeld, and R. Armiento, International Journal of Quantum Chemistry **115**, 1094 (2015).
[15] L. Ward, R. Liu, A. Krishna, V. I. Hegde, A. Agrawal, A. Choudhary, and C. Wolverton, Phys. Rev. B Condens. Matter **96**, 024104 (2017).
[16] C. Oses, C. Toher, E. Gossett, A. Tropsha, O. Isayev, and S. Curtarolo, Nature communications **8**, 1 (2017).
[17] K. T. Schütt, F. Arbabzadah, S. Chmiela, K. R. Müller, and A. Tkatchenko, Nat. Commun. **8**, 13890 (2017).
[18] K. T. Schütt, H. E. Sauceda, P.-J. Kindermans, A. Tkatchenko, and K.-R. Müller, J. Chem. Phys. **148**, 241722 (2018).
[19] T. Xie and J. C. Grossman, Physical Review Letters **120**, 145301 (2018).
[20] P. Zaspel, B. Huang, H. Harbrecht, and O. A. von Lilienfeld, Journal of Chemical Theory and Computation , acs.jctc.8b00832 (2019).

[21] B. Meredig, A. Agrawal, S. Kirklin, J. E. Saal, J. W. Doak, A. Thompson, K. Zhang, A. Choudhary, and C. Wolverton, Phys. Rev. B Condens. Matter **89**, 094104 (2014).

[22] A. Jain and T. Bligaard, Phys. Rev. B Condens. Matter **98**, 214112 (2018).

[23] S. J. Chung, T. Hahn, and W. E. Klee, Acta Crystallogr. A **40**, 42 (1984).

[24] H. J. A. M. Heijmans and A. V. Tuzikov, IEEE Transactions on Pattern Analysis and Machine Intelligence **20**, 980 (1998).

[25] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, in *International Conference on Machine Learning* (2017) pp. 1263–1272.

[26] P. B. Jørgensen, K. W. Jacobsen, and M. N. Schmidt, (2018), arXiv:1806.03146 [stat.ML].

[27] K. He, X. Zhang, S. Ren, and J. Sun, (2015), arXiv:1512.03385 [cs.CV].

[28] A. Jain, S. P. Ong, G. Hautier, W. Chen, W. D. Richards, S. Dacek, S. Cholia, D. Gunter, D. Skinner, G. Ceder, and K. a. Persson, APL Materials **1**, 011002 (2013).

[29] G. Bergerhoff, R. Hundt, R. Sievers, and I. D. Brown, J. Chem. Inf. Comput. Sci. **23**, 66 (1983).

[30] J. E. Saal, S. Kirklin, M. Aykol, B. Meredig, and C. Wolverton, JOM **65**, 1501 (2013).

[31] S. Kirklin, J. E. Saal, B. Meredig, A. Thompson, J. W. Doak, M. Aykol, S. Rühl, and C. Wolverton, npj Computational Materials **1**, 15010 (2015).

[32] A. H. Larsen, J. J. Mortensen, *et al.*, Journal of Physics: Condensed Matter **29**, 273002 (2017).

[33] J. Enkovaara, C. Rostgaard, J. J. Mortensen, *et al.*, Journal of Physics: Condensed Matter **22**, 253202 (2010).

[34] J. P. Perdew, A. Ruzsinszky, G. I. Csonka, O. A. Vydrov, G. E. Scuseria, L. A. Constantin, X. Zhou, and K. Burke, Phys. Rev. Lett. **100**, 136406 (2008).

[35] J. P. Perdew, K. Burke, and M. Ernzerhof, Phys. Rev. Lett. **77**, 3865 (1996).

[36] H. J. Monkhorst and J. D. Pack, Phys. Rev. B **13**, 5188 (1976).

[37] D. Kingma and J. Ba, (2014), arXiv:1412.6980 [cs.LG].

[38] W. E. Klee, Crystal Research and Technology **39**, 959 (2004).

[39] N. W. A. Gebauer, M. Gastegger, and K. T. Schütt, (2018), arXiv:1810.11347 [stat.ML].

[40] G. Thimm, Acta Crystallogr. A **65**, 213 (2009).

[41] J.-G. Eon, Acta Crystallogr. A **67**, 68 (2011).

[42] A. A. Peterson, R. Christensen, and A. Khorshidi, Phys. Chem. Chem. Phys. **19**, 10978 (2017).

[43] B. Lakshminarayanan, A. Pritzel, and C. Blundell, in *Advances in Neural Information Processing Systems 30*, edited by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Curran Associates, Inc., 2017) pp. 6402–6413.

[44] T. Pearce, N. Anastassacos, M. Zaki, and A. Neely, (2018), arXiv:1805.11324 [stat.ML].

[45] I. Osband, J. Aslanides, and A. Cassirer, (2018), arXiv:1806.03335 [stat.ML].

APPENDIX C

Deep Generative Models for Molecular Science

The following paper is published as,

Peter B. Jørgensen, Mikkel N. Schmidt, and Ole Winther. 2018. "Deep Generative Models for Molecular Science." Molecular Informatics 37 (1-2).

Minireview

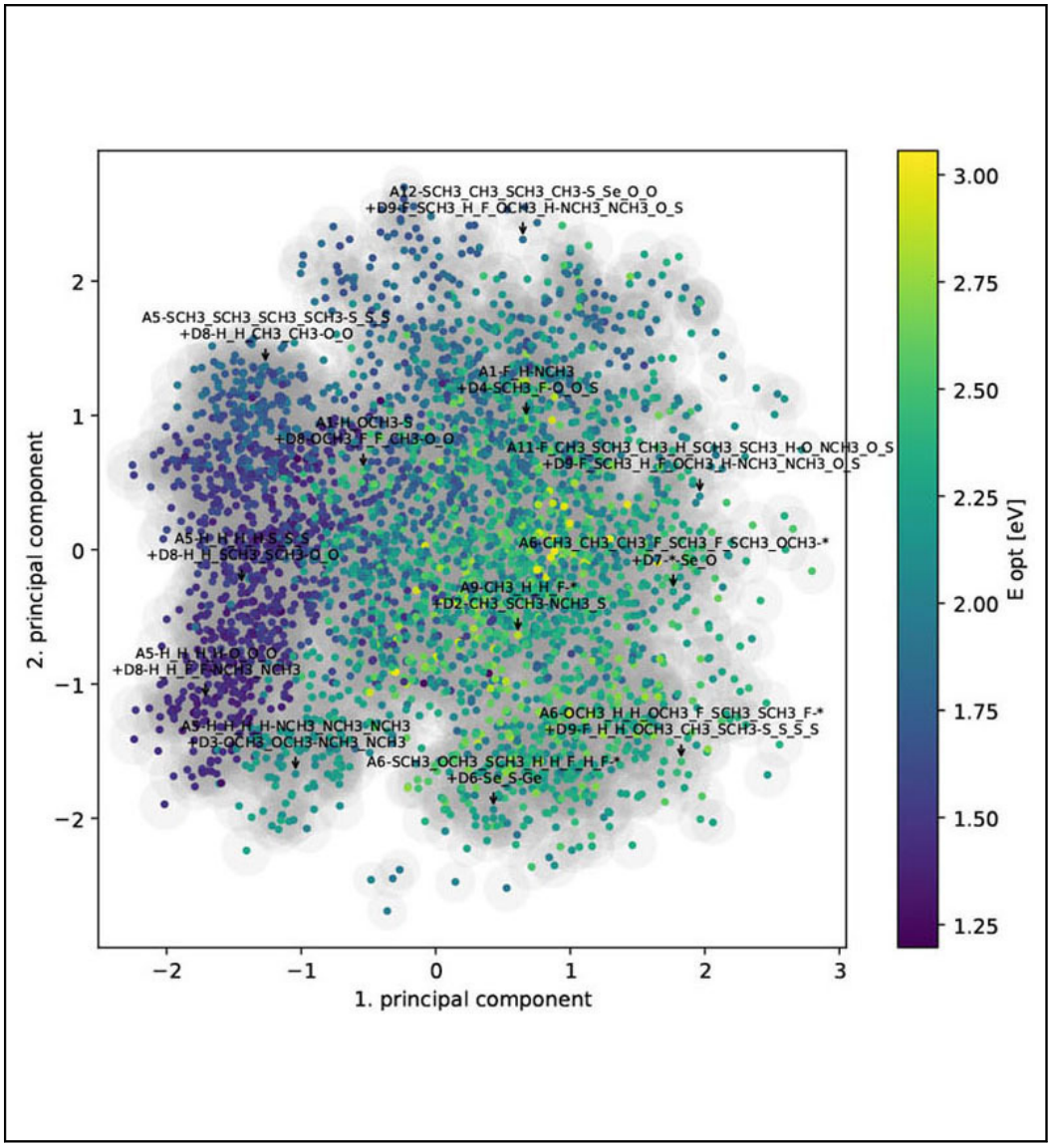
www.molinf.com

molecular informatics

DOI: 10.1002/minf.201700133

Deep Generative Models for Molecular Science

Peter B. Jørgensen,[a] Mikkel N. Schmidt,[a] and Ole Winther*[a]



© 2018 Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim

Abstract: Generative deep machine learning models now rival traditional quantum-mechanical computations in predicting properties of new structures, and they come with a significantly lower computational cost, opening new avenues in computational molecular science. In the last few years, a variety of deep generative models have been proposed for modeling molecules, which differ in both their model structure and choice of input features. We review these recent advances within deep generative models for predicting molecular properties, with particular focus on models based on the probabilistic autoencoder (or variational autoencoder, VAE) approach in which the molecular structure is embedded in a latent vector space from which its properties can be predicted and its structure can be restored.

Keywords: molecular science · deep learning · variational inference · variational auto-encoders · generative modeling

1 Introduction

Computational molecular science – the discovery and design of new molecules and analysis of their structure and properties by computer models – has traditionally involved making elaborate quantum-mechanical computations derived from first principle. In recent years, new approaches based on machine learning have shown great promise, approaching the same accuracy as first principle computations at a much lower computational cost. Machine learning is a branch of artificial intelligence concerned with making models that can learn from data by discovering patterns in the data, and generalizing these patterns to new unseen cases. In molecular science, machine learning can leverage the existing huge databases of experimental results and quantum-mechanical calculations that are currently available, to learn to predict properties and structures of new molecules at unrivaled computational speed. In this review we outline the current trends in machine learning-based computational molecular science with a particular focus on one of the most promising model classes known as *deep generative models*. Based on the latest results from the literature and from our own research, our aim is to characterize the deep generative modeling paradigm in terms of both model structure and approach to inference, in order to build intuition about the mechanisms behind its success.

In machine learning we distinguish between discriminative and generative models. In a *discriminative* learning approach to molecular science, we would be concerned with learning a mapping from a molecule x to a property y that we are interested in predicting. Given a dataset $\{x_i, y_i\}_{i=1}^N$ that consists of N molecules and their corresponding known properties, the discriminative model is targeted at learning the probability distribution $p(y|x)$ such that predictions for a new material x^* can be computed. Although this can lead to excellent predictions, the downside of the discriminative approach is that the model does not describe the molecular structure x itself – only the relation between the structure and property – and can thus not directly be used to make inference about new molecules of interest. In a *generative* learning approach, the model is concerned with characterizing either the distribution of the molecular structure $p(x)$ or the joint distribution $p(x, y)$ of the molecular structure and the property of interest. The former case where only the molecular structure is modeled is known as *unsupervised* learning, whereas the latter case where both molecular structure and corresponding properties are modeled is known as *supervised* learning.

In this study we will focus on one particular non-linear flexible generative model, the *variational autoencoder (VAE)*.[1,2] The complementary likelihood free approach to generative modelling, *generative adversial networks (GAN)*, has also received a lot of attention recently. GAN and other approaches beyond VAE are discussed in Section 6. In our review of VAE we include recent advances in inference and implementation details. Therefore the tutorial by Doersch[3] might be an easier starting point to establish the mathematical intuition behind the model.

The remainder of the paper discusses generative modelling (Section 2), variational inference (Section 3), practical implementation (Section 4) and application to properties of molecules (Section 5).

2 Deep Generative Models

In this paper we define a generative model as a probabilistic model for the data features we have access to. We distinguish between unsupervised learning where we have a feature vector x and supervised learning where we have x and dependent variables y. For unsupervised learning the generative model is the joint distribution of x: $p(x)$. Probabilistic supervised learning involves modeling the conditional $p(y|x)$. In generative probabilistic supervised learning we decompose the joint distribution of x and y as $p(x, y) = p(y|x)p(x)$. In the molecular context x will usually be some representation of the molecule structure (for example the SMILES representation,[4] molecular fingerprints such as MACCS[5] or in principle the atomic positions and properties) and y will be a physical property of the system such as free energy, ground state energy, band gap, or crystal structure

[a] *P. B. Jørgensen, M. N. Schmidt, O. Winther*
Technical University of Denmark
E-mail: olwi@dtu.dk

 © 2018 Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim

or a biological response such as toxicity, cellular uptake, or drug efficacy.

The two main reasons why we want to apply probabilistic generative models are that they:

1. Allow for quantitative model comparision. Once fitted we can evaluate the density on a test data point x_{test} or a test dataset. A high value of log $p(x_{\text{test}})$ (relative to other model baselines) indicates that the model has captured essential statistical properties of x.
2. Can be used to synthesize new data. Once fitted we can simulate new data from the model $x_{\text{new}} \sim p(x)$ where $\sim$ is notation for a draw from the distribution. The synthesised data can be used for a qualitative evaluation of the model for example an image generated from a model fitted on a training set of natural images. The generated data can also be of practical interest, for example a new molecule with potentially desirable properties.

In this paper we will consider a specific class of generative models called latent variables models. A latent variable (vector) z represents unobserved properties of the datum that can describe the observed datum x statistically through a generative process $p(x|z)$. A simple example of such a model is a linear model with additive noise ε: $x = Wz + \varepsilon$, where W is the weight matrix. Often we will assume that $\dim(x) > \dim(z)$ so that the latent representation is more compact than the observed data. The latent variable itself is assumed to be generated from a prior distribution $p(z)$. The generative model $p(x)$ discussed above is recovered by marginalizing over $z : p(x) = \int p(x|z)p(z)dz$. In a latent variable model we can draw from $p(x)$ by a two-step procedure: 1) $z \sim p(z)$ and 2) $x \sim p(x|z)$. In some cases the latent variable will have a direct physical meaning and in other cases the use of a latent variable model is a convenient way to define a flexible statistical model.

We will usually use maximum likelihood learning. For unsupervised learning this means that we have a training set $X = \{x_i\}_{i=1}^N$ that we model with a latent variable model $p_\theta(x, z) = p_\theta(x|z)p(z)$ with parameters θ. We will as a default assume independent identically distributed (iid) samples such that the likelihood for θ is written as

$$p_\theta(x) = \prod_{i=1}^{N} p_\theta(x_i) \tag{1}$$

with $p_\theta(x_1) = \int p_\theta(x_i|z)p_\theta(z)dz$. Sometimes we will omit the θ dependence for brevity. In maximum likelihood learning the objective is thus to maximize (1) with respect to the parameters θ.

In deep generative modelling we replace the simple linear relation between the observed data x and the latent variable z with a parameterized non-linear function $f_\theta(\cdot)$. For $f_\theta(\cdot)$ we use a multi-layered neural network with L layers of adaptable weights, for example

$$f_\theta(z) = W_L h_{L-1} + b_L, \tag{2}$$

$$h_l = \text{relu}(W_l h_{l-1} + b_l),\ l = 2, \ldots, L-1 \tag{3}$$

$$h_1 = \text{relu}(W_1 z + b_1), \tag{4}$$

where the (element-wise) rectified linear activation function is given by $\text{relu}(a) = \max(0, a)$. We then have

$$p_\theta(x|z) = N(f_\theta(z), \sigma^2 I). \tag{5}$$

The trainable parameters θ of the model are the L weight matrices $W_1, \ldots W_L$, the L bias vectors $b_1, \ldots b_L$ and the output noise variance σ^2. The model can be extended, for example by letting σ^2 depend on the latent variable z or by introducing a hierarchy of latent variables, for example $p(x, z_1, z_2) = p(x|z_1)p(z_1|z_2)p(z_2)$ where z_1 and z_2 are latent variables vectors.

Peter B. Jørgensen received his B.Sc. (2011) in Electronics Engineering and IT and his M.Sc. (2013) in Wireless Communication Systems from Aalborg University, Denmark. He is currently working on his Ph.D. in Machine Learning at Technical University of Denmark. PBJs research interests includes (variational) Bayesian inference, statistical signal processing and deep learning.

Mikkel N. Schmidt is Associate Professor at DTU Compute, Technical University of Denmark. He is interested in probabilistic modeling and statistical machine learning with applications in both science and industry. His primary focus is the development of computational procedures for inference and validation.

Ole Winther is Professor of Data Science and Complexity at the DTU Compute, Technical University of Denmark. OWs Research interests include general machine learning methodology, deep generative modeling and applications of machine learning in health- and bioinformatics, energy and natural language.

 © 2018 Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim

As soon as we introduce non-linearities in the model we can not analytically marginalize out the latent variables. We therefore have to resort to approximations. A prominent method, that variational autoencoders are based on, is variational inference.

3 Variational Inference

The approach we use in variational autoencoders (VAE) consists of three steps:

Likelihood lower bound. Replace the likelihood with a more tractable lower bound. The most widely used is:

$$\log p(x_i) \geq E_{q_i(z)}\left[\log \frac{p(x_i|z)p(z)}{q_i(z)}\right], \tag{6}$$

where $E_{q_i(z)}[...]$ denote average (expectation) over $q_i(z)$. We have here introduced a variational distribution $q_i(z)$ which is an approximation to the posterior distribution over the latent variables and its role in the variational autoencoder is explained in the next paragraph. This lower bound still requires intractable integration due to $E_{q_i(z)}[...]$, but we can derive low variance Monte Carlo estimators for these. The bound can be decomposed into two terms

$$\log p(x_i) \geq E_{q_i(z)}[\log p(x_i|z)] - KL(q_i(z), p(z)), \tag{7}$$

where $KL(q(x), p(x)) \equiv \int q(x) \log \frac{q(x)}{p(x)} dx$ is the Kullback-Leibler divergence between q and p. The first term on the right-hand-side of (7) can be interpreted as the average reconstruction error, i.e. how well does the generative model fit the data distribution and the KL-term is a measure of how much $q(z)$ diverges from the prior $p(z)$.

An alternative tighter bound is given by Burda et al.,[6] which is often used when comparing different models on the same dataset.

$$F_K(x) \equiv \int \prod_{k=1}^{K} q(z_k) \log\left[\frac{1}{K}\sum_{k'=1}^{K} w_x(z_{k'})\right] \prod_{k=1}^{K} dz_k$$

with $w_x(z) \equiv \frac{p(x|z)p(z)}{q(z)}$. This so-called importance weighted bound coincides with the standard bound for $K=1$, obeys $F_K \geq F_L$ for $K > L$ and converges to the log $p(x)$ for $K \to \infty$ as a consequence of the law of large numbers. It is often used with moderate K for example $K \approx 3$–10 during training and large K, $K \approx 1000$–5000 for evaluating the test log likelihood.

Inference network. The lower bound (6) depends upon a variational distribution $q_i(z)$ which is an approximation to the latent posterior distribution

$$p(z|x_i) = \frac{p(x_i|z)p(z)}{p(x_i)}. \tag{8}$$

We can for example choose $q_{\phi_i}(z) = N(z|\mu_i, \sum_i)$ with variational parameters $\phi_i = \{\mu_i, \Sigma_i\}$. In VAE, instead of having a set of parameters for each x_i we will use a socalled inference network parameterization $q_\phi(z|x)$. The inference network will be specified as a deep neural network similar to the generative model but now with x as input and mean and variance of z as output. We could for example take $q_\phi(z|x) = N(z|\mu_\phi(x), \Sigma_\phi(x))$ where $\mu_\phi(x)$ and $\Sigma_\phi(x)$ are written as output of a deep network in the same fashion as the generative model was specified in the generative model Eqs. (2)–(5). This is actually more restrictive than the formulation with individual variational parameter for each datum but it has the advantage that we leverage information across data points. In other words it is based upon the assumption that data points that are close (in some sense derived from the data) will also have similar posterior latent distribution. The log likelihood lower bound for the training set that we want to optimize is:

$$L(\theta, \phi) = \sum_{i}^{N} E_{q_\phi(z|x_i)}\left[\log \frac{p_\theta(x_i|z)p_\theta(z)}{q_\phi(z|x_i)}\right]. \tag{9}$$

Note that even though we treat the two sets of parameters θ and ϕ in the same way they play different roles: the θ-optimization is model fit and the ϕ-optimation is for making the bound as tight as possible. Overfitting is not an issue in the latter case – we want the inference network to come as close as possible to $p(z|x)$.

The inference network $q_\phi(z|x)$ can be interpreted as a probabilistic encoder which maps an input x to a probability distribution in the latent vector space and $p_\theta(x|z)$ is the corresponding probabilistic decoder. Hence the name *variational autoencoder*.

Monte Carlo estimators and reparameterization trick. Low variance Monte Carlo estimators of the log likelihood lower bound and its derivative with respect to the parameters θ and ϕ are obtained by choosing a parameterization of $q_\phi(z|x)$ that allows the use of the so-called reparameterization trick. Again we focus on the standard bound and use M samples:

$$\log p(x) \geq E_{q_\phi(z|x)}[\log w_x(z)] \approx \frac{1}{M}\sum_{m=1}^{M} \log w_x(z^m)$$

with $w_x(z) = \frac{p_\theta(x|z)p_\theta(z)}{q_\phi(z|x)}$ and $z^m \sim q(z|x)$. Since we expect $w_x(z)$ to scale exponentially in the number of dimensions of z then the logarithm appearing in the bound is important to make the Monte Carlo estimator (the right hand side above) have low variance.

We need the reparameterization trick when taking derivatives. For the Gaussian inference network it amounts to replacing an average over $q_\phi(z|x)$ with an average over $\varepsilon \sim N(0, I)$. The parameter dependence is thus shifted into the integrand. In the integrand we will then replace z with $z_\phi(\varepsilon, x) = \mu_\phi(x) + \sqrt{\sum_\phi(x)}\varepsilon$ where $\sqrt{\sum_\phi(x)}$ is to be under-

 © 2018 Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim

stood as a matrix square root. A diagonal $\Sigma_\phi(x)$ is the standard choice because it is simple only requiring a network with dim(z) outputs for the (log) variances.

3.1 Bits Back

The bits back argument[7,8,9] gives theoretical insight on why variational models often will converge to a solution where relatively few latent dimensions are being used. That is in $p(x|z)$ many of the components of z have no influence on x. The bits back back argument is simply an alternative decomposition of the log likelihood bound in the limit of very large training set corresponding to averaging over the data generating distribution $p_{\text{data}}(x)$. For the variational bound using an inference network we have

$$E_{p_{data}(x)}[\log p(x)] \geq E_{p_{data}(x),q(z|x)}\left[\log \frac{p(x|z)p(z)}{q(z|x)}\right].$$

We can rewrite this expression using two steps: multiplying by $\frac{p_{data}(x)}{p_{data}(x)}$ inside the log and replacing $p(x|z)p(z)$ by the equivalent $p(z|x)p(x)$:

$$E_{p_{data}(x)}[L] = -H(p_{data}) - KL(p_{data}(x), p(x)) \\ -E_{p_{data}(x)}[KL(q(z|x), p(z|x))],$$

where $H(p) = -\int p(x) \log p(x) dx$ is the entropy of p. The first term (minus the entropy of the data generating distribution) is the irreducible lower bound on the log likelihood, the second is the model error and the third is the variational posterior approximation error. The trade-off between the two last terms determines the solution we will find. We get a more flexible model by introducing latent variables because $p(x) = \int p(x|z)p(z)dz$ will in general be more flexible than a model without the latent variables but we pay a price in terms of the variational approximation error. The components $\hat{z}$ of z not are used in the generative model will have $p(\hat{z}|x) = p(\hat{z})$ and we can get $KL(q(\hat{z}|x), p(\hat{z}|x))$ to be zero by setting $q(\hat{z}|x) = p(\hat{z})$. The limited number of active units observed empirically, see for example,[10] reflects this trade-off. At some point introducing additional latent variables gives a smaller gain in terms of model fit than the price paid in approximation error. This is motivation for using the improved variational approximations discussed in the text section.

3.2 Improving the Variational Approximation

Methods for improving the variational distribution mayroughly be divided into three categories:

Hierarchical specification of variational distribution. For generative models with two or more stochastic layers there is some freedom in choosing the connectivity of variational distribution. Consider a two layer generative model $p(x, z_1, z_2) = p(x|z_1)p(z_1|z_2)p(z_2)$. One minimal solution is to specify q with dependence in the reverse order: $q(z_1, z_2|x) = q(z_2|z_1)p(z_1|x)$. However, we get a more flexible and thus more accurate inference network if we condition the first term on x as well: $q(z_1, z_2|x) = q(z_2|z_1, x)p(z_1|x)$. We say that we introduce a skip-connection that connects x directly to the network for the mean and covariance of z_2: $\mu_\phi(z_1, x)$ and $\Sigma_\phi(z_1, x)$. The ladder VAE[10] has a more advanced version of skip connections which also includes a parameterization where the parts of the inference network are shared by the prior hierarchical prior specification for the generative model. This leads to improved generative performance.

Normalizing flows. Let f be an invertible dim(z)→dim(z) mapping. We can use distribution $z' = f(z)$ in our generative model instead of z and introduce the determinant of the Jacobian in the likelihood.[11] If we let f have adaptable parameters it is possible to learn quite flexible priors. We can also generalize this concept in a simple way by letting f be a series on invertivle transformations.[12,13]

Auxiliary latent variables. In the auxiliary variable approach[14,15,16] we introduce a new latent vector a into the inference network $q(z, a|x) = q(z|a, x)q(a|x)$. We specify the generative model as $p(x, z, a) = p(x|z)p(z)p(a|z, x)$ in order to leave the generative process unaffected that is z and x are not affected by the value of a. However, the resulting marginal inference network $q(z|x) = \int q(z|a, x)q(a|x)da$ is now more flexible than before and should therefore give a better fit to the posterior. This is also observed in practice.[15] One may also make a bits back argument for the auxiliary model that shows that we still have to make a trade-off between the exact posterior $p(a|z, x)$ and its variational approximation.

4 VAE Inference in Practice

Variational autoencoder practice follows standard modern deep learning practice. Mini-batch stochastic gradient descent optimization is employed with parameter updates based upon gradients calculated on mini-batches of the order of 100 training examples. Thousands of epochs (complete sweeps through the training set) are often needed for convergence. The objective is usually evaluated with $M = 1$ samples for the expectation:

$$\sum_i^{\text{mini-batch}} \log \frac{p_\theta(x_i|z_i)p_\theta(z_i)}{q_\phi(z_i|x_i)}$$

with $z_i \equiv z_\phi(\varepsilon_i, x_i) = \mu_\phi(x_i) + \sigma_\phi(x_i) \otimes \varepsilon_i$, $\mu_\phi(x_i)$ and $\log \sigma_\phi(x_i)$ are the output of the inference network (both dim(z) dimensional), $\otimes$ denotes component-wise multiplication and $\varepsilon_i \sim N(\varepsilon|0, I)$. Setting $M = 1$ is the favored choice because averaging over mini-batches will be a more efficient way to decrease the variance of the estimator than to use the same example $M > 1$ in the same mini-batch. Many variants of

 © 2018 Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim

adaptive step-size stochastic gradient descent with momentum have been proposed recently. Adam[1] is a popular choice. Differentiation of the objective involves applying the chain rule of differentiation. This is performed automatically within modern deep learning software packages such as TensorFlow and PyTorch. Some derivate terms in the objective cancels exact on expectation.[17] These terms have to removed explicitly from the gradient calculation. In practice it turns out that it does not make a big difference to take this into account or not.[17]

The effect of removal of latent variables discussed in Section 3.1 can be partially mitigated by employing "warm-up"[10,18] or the "free bits" surrogate objective.[13] We rewrite the log likelihood lower bound objective in (9) and introduce a "temperature" parameter α:

$$L(\theta,\phi,\alpha)=\sum_{i}^{N} E_{q_\phi(z|x_i)}\left[\log p_\theta(x_i|z)+\alpha\log\frac{p_\theta(z)}{q_\phi(z|x_i)}\right]. \qquad (10)$$

The original objective is restored by setting $\alpha=1$. Warmup[10,18] amounts to starting with a traditional map x-to-x autoenocder objective corresponding to $\alpha=0$ and slowly increasing α, say linearly during the first 200 epochs, to the variational objective $\alpha=1$. In some cases for example in the sequence encoder and decoder models employed in Section 5 it turns out that it is necessary to set the final value of α below one in order to make the model use any latent variables. The learned latent structure is still useful but the likelihood lower bound interpretation is no longer valid.

5 Applications to Molecular Sciences

Using VAEs with recurrent neural network encoder and decoder was first employed by Bowman et al.[18] to generate English written sentences. The first application in molecular science is demonstrated by Gómez-Bombarelli et al.,[19] where a similar encoder and decoder model is used to generate SMILES strings character by character from the latent space. The method is applied to a dataset of approximately 250,000 drug-like molecules from the ZINC database[20] and 100,000 organic light emitting diode (OLED) molecules.[21] One of the problems of the character-based VAE is that it often produces invalid molecules. In the experiments by Gómez-Bombarelli et al.[19] from 70% to less than 1% of the generated samples are valid molecules. The errors can be syntax errors (the generated string is not a valid SMILES string) or semantic errors (the SMILES string is syntactically valid but the molecule corresponding to the SMILES string is physically impossible). Despite these problems, using SMILES strings is an appealing approach because it gives a full description of the molecule without making assumptions about which features are important for the task at hand. If we for example modeled fingerprints we would have to search for the molecule that corresponds to the generated fingerprint (which might not exist if it is generated erroneously) and we have already made a choice of which features are important by selecting or designing the fingerprint.

5.1 Using a Grammar

The number of syntax errors can be significantly reduced by replacing the character-based encoder and decoder with a syntax-aware model as done in the Grammar VAE.[22] The SMILES syntax can approximately[1] be described by a so-called context free grammar. Rather than constructing a string character by character we can be represent the string as a sequence of production rules. Not all possible sequences of production rules are valid and the formulation as a context free grammar allows us to enforce this restriction upon the decoder such that all generated sequences are syntactically valid. Kusner et al.[22] also applies the Grammar VAE model to the drug-like molecules from the ZINC database and the results indicates improved smoothness in the latent space representation in comparison to the character based VAE.

In many molecular screening applications we are not interested in the set of all possible molecules, but for example want to limit our search to a smaller set of molecules that are easy to synthesize. In this case the SMILES grammar formulation may be unnecessarily complex and can be replaced with a simpler application specific grammar that is easier to handle for the grammar VAE. This approach is used by Jørgensen et al.[23] for screening of materials for polymer solar cells where each material is composed from a library of acceptor, donor and side group substructures. If the application allows it, the grammar can be formulated such that a syntactically valid string implies a semantically valid molecule, such that the model only generates valid molecules.

5.2 Example: Screening of Polymer Solar Cells Using Grammar VAE

The problem of interest in[23] is to find new materials for polymer solar cells. The polymer units are composed by one of 13 acceptor units, one of 10 donor units and a number of side groups. The crucial properties are the Lowest Unoccupied Molecular Orbital (LUMO) and the optical band gap energy. These properties can be estimated with computationally costly DFT calculations and from a dataset of approximately 4000 DFT calculations we seek to propose new candidate molecules and determine their properties with machine learning. An example polymer solar cell

[1] Not all elements of SMILES are context free, e.g. opening and closing of ring-bonds where the same digit must be used for opening and closing the bond as seen in benzene "c1ccccc1".

 © 2018 Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim

molecule (monomer) is shown in Figure 1 and the context free grammar that describes all the possible strings is shown in Figure 2. A model based on this grammar might generate invalid molecules because the number of side groups for each acceptor/donor is not defined by the grammar, but the grammar rules are simpler this way.

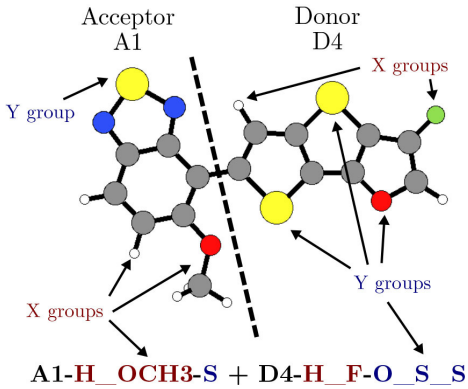


Figure 1. An example molecule from the polymer solar cell dataset and its simplified string representation.

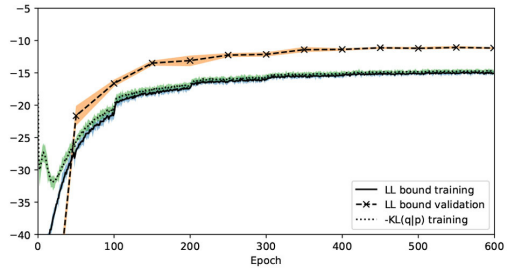
```
m      →  A – OPTG – OPTG + D – OPTG – OPTG
A      →  A1 | A2 | A3 | A4 | A5 | A6 | A7 | A8 | A9 | A10 | A11 | A12 | A1
D      →  D1 | D2 | D3 | D4 | D5 | D6 | D7 | D8 | D9 | D10
OPTG   →  * | GS
GS     →  G | GS _ G
G      →  Ge | CH3 | OCH3 | H | C | O | SCH3 | NCH3 | S | F | Si | Se
```

Figure 2. Context free grammar for the polymer solar cell dataset.

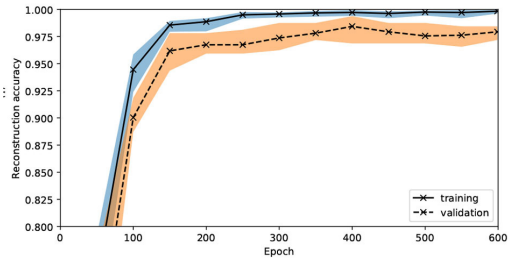
We train a grammar VAE using the log likelihood lower bound objective (10) with a constant temperature parameter $\alpha < 1$. Initial studies showed that using $\alpha < 0.1$ was necessary in order to yield good reconstruction error and we use $\alpha = 0.08$ for the training objective and $\alpha = 1$ when evaluating the bound. We use Adam[24] with initial learning rate 0.001 and the learning rate is divided by 2 after every 100 epochs. The log likelihood lower bound for the training and validation set is shown in Figure 3a and the reconstruction accuracy[2] is shown in Figure 3b.

The latent space dimension is set to 32, but as discussed in Section 3.1 it might not be beneficial to use all the latent variables. Because the prior $p(z)$ and $q(z|x)$ factorize across the dimensions of z we can compute the KL-term of the objective function for each of the latent dimensions as shown in Figure 4 for the polymer solar cell grammar VAE. In this example the "effective" dimension of the latent space is around 17, the KL-term for the remaining dimensions is close to zero. Depending on the subsequent application of the model, it may be beneficial to "prune" these dimensions from the model.

The embedding is visualized using principal component analysis in Figure 5. Even though the grammar VAE is



(a) Log likelihood lower bound and KL-term versus training epoch. The validation lower bound is higher than the training bound, because we use the tighter importance weighted bound with $K = 50$ samples for validation and $K = 1$ for training.



(b) Average reconstruction accuracy.

Figure 3. Training curves for grammar VAE on the polymer solar cell dataset. We use 5-fold cross-validation to estimate the model's generalisation error. The lines are averages across the 5 folds and the shaded area shows the maximum and minimum.

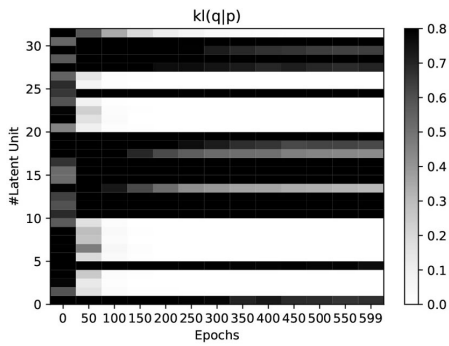
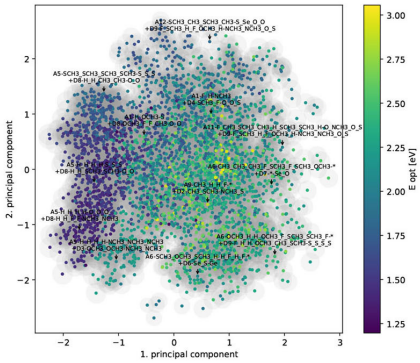


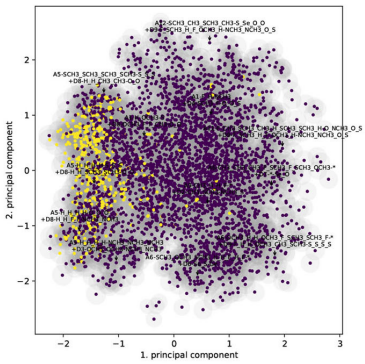
Figure 4. Average (over the training set) KL-divergence

[2] The reconstruction accuracy is measured in the following way: Each string x is encoded as the mean of the encoding distribution $q_\phi(z|x)$ and decoded using $p_\theta(x|z)$ where the most probable (according to the decoder model) production rule is selected at each step and the encode/decode is considered successful if the output of the decoder matches the encoder input exactly. The accuracy is the success rate across the data set.

 © 2018 Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim



(a) Embedding coloured according to optical gap value



(b) Embedding coloured according to whether the LUMO and optical gap energies both are within the target range. The yellow points are the "good" molecules

Figure 5. Visualization of embedding using principal component analysis on the mean value of the embedding corresponding to each data point. The dots marks the mean value and the shaded ellipses are the contours corresponding to half the standard deviation of each point.

trained without using label information we notice some structure when coloring the embedded data points according to the optical band gap energy. The target range for the optical gap is from 1.4 eV to 1.7 eV and the LUMO target range is −3.2 eV to −2.9 eV.

5.3 Generating New Molecules

In our description above we have only discussed the problem of encoding to and decoding from a latent space representation of the discrete input, but often we would like to use this representation to search for molecules with a given property. Since the latent space is a continuous vector space we can then employ our favourite regression/classification on top of this representation. Kusner et al.[22] and Gómez-Bombarelli et al.[19] use a Gaussian process regression model that is trained on the latent space representation and new points in the latent space are selected based on Bayesian optimization, see Brochu et al.[25] for a tutorial on Bayesian optimization. The Bayesian optimization procedure tries to avoid to sample new points that are close to the training data, so we can avoid sampling a point in the latent space that decodes to a molecule that is already in the training set. However, this also means we might sample from areas in the latent space that decodes to invalid molecules.[19] Instead they train a feedforward neural network on the latent representation and then optimize the position by taking a few gradient steps starting from the latent representation of a molecule with good properties.[19] Jørgensen et al.[23] continuing the example above) sample a large number of random points from the latent space and this set of points is decoded and encoded repeatedly to find well-behaved regions of the latent space. After a number of iterations the best points according to the regression model are selected for further studies.

A more advanced approach is to train a regression model in conjunction with the VAE model, as done for classification in semi-supervised learning with deep generative models.[26] In this class of models we can condition the generation of new samples on a specific class label. This is yet to be tried with the SMILES representation, but has been successfully done for drug efficiency classification trained on preand post-treatment gene expression vectors.[27]

6 Other Generative Models

Another popular deep generative model class is Generative adversarial networks (GAN).[28] In this framework the generative model is pitted against a discriminative adversarial model, which is typically also implemented as a neural network. During training the discriminator is optimized to classify whether a sample comes from the data distribution or it is generated by the generative model and the generative model is optimized to fool the discriminator. Some advantages compared to VAEs is that we do not need to specify a variational distribution $q_\phi(z|x)$ and the model can represent sharper data distributions.[28]

The GAN framework has been applied to generation of molecular fingerprints in the DruGAN Adversarial Autoencoder.[29] They effectively replace the KL-term of a VAE with a discriminator that discriminates between the encoder and a sample from $N(0, \mathbf{I})$. Apart from generating new molecular fingerprints they use the VAE as pretraining for an aqueous solubility regression problem.

It is also possible to formulate generative models based on (deep) reinforcement learning in which a software agent builds a data point through a series of actions, e.g. selecting the characters for a SMILES string representation. The agent is trained to maximize a given notion of reward for the generated data. The advantage of this approach is that the reward function can be any function of the taken actions as opposed to VAE where we need to specify a differentiable $p(x|z)$ and for a GAN the discriminator must be differentiable.

 © 2018 Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim

Olivecrona et al.[30] trains an autoencoder to reconstruct SMILES strings. Then they replace the decoder with a reinforcement-learning trained decoder to generate molecules with a desired property. They use a trade-off parameter to trade off between generating from the autoencoder (prior) or follow the RL cost function. In objective-reinforced generative adversarial network (ORGAN),[31] the GAN framework is combined with reinforcement learning to generate SMILES strings. They also introduce a trade-off parameter that weights between GAN training (to make molecules look like training data) and RL training (to make molecules with desired property).

7 Conclusion

This paper has given an introduction to variational autoencoders (VAE) and given a few examples of their application for modeling molecule properties. VAE are flexible non-linear latent variables generative models. At the time of writing the VAE have only been around for four years. During these years variants of VAE have pushed the state-of-the-art performance in many unsupervised and semi-supervised benchmarks and found its way into many application areas. Un- and semi-supervised learning are arguably areas where we will see much more research focus in the coming years because supervised learning is much more explored and better understood and because having access to good unsupervised models will enable researchers to explore the vast amounts of unlabeled data available. There are still many open issues around inference with VAE such as how to construct better variational approximations (inference networks) such that we can learn better model for for example sequence data. When these current shortcomings have been dealt with we will likely see even more applications in many data abundant areas such as computational materials science and biomedicine.

Conflict of Interest

None declared.

References

[1] M. Kingma, Diederik P; Welling, *arXiv preprint arXiv:1312.6114* **2013**, 1312.6114.
[2] D. J. Rezende, S. Mohamed, D. Wierstra, *arXiv preprint arXiv:1401.4082* **2014**, 1401.4082.
[3] C. Doersch, *arXiv preprint arXiv:1606.05908* **2016**, 1606.05908.
[4] D. Weininger, *J. Chem. Inf. Comput. Sci.* **1988**, *28*, 31–36.
[5] MDL Information Systems. MACCS Keys.
[6] Y. Burda, R. Grosse, R. Salakhutdinov, In *Proceedings of the International Conference on Artificial Intelligence and Statistics*.
[7] G. E. Hinton, D. Van Camp, In *Proceedings of the sixth annual conference on Computational learning theory*, ACM, pp. 5–13.
[8] A. Honkela, H. Valpola, *IEEE Transactions on Neural Networks* **2004**, *15*, 800–810.
[9] X. Chen, D. P. Kingma, T. Salimans, Y. Duan, P. Dhariwal, J. Schulman, I. Sutskever, P. Abbeel, In *International Conference on Learning Representations*.
[10] C. K. Sønderby, T. Raiko, L. Maaløe, S. K. Sønderby, O. Winther, In *Advances in Neural Information Processing Systems 29*, **2016**.
[11] D. J. Rezende, S. Mohamed, *arXiv preprint arXiv:1505.05770* **2015**.
[12] L. Dinh, D. Krueger, Y. Bengio. *arXiv preprint arXiv:1410.8516* **2014**.
[13] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, M. Welling, *arXiv preprint, arXiv:1606.04934* **2016**, 1606.04934.
[14] F. Agakov, D. Barber, In *Neural Information Processing, vol. 3316 of Lecture Notes in Computer Science*, Springer Berlin Heidelberg, **2004**, pp. 561–566.
[15] L. Maaløe, C. K. Sønderby, S. K. Sønderby, O. Winther. In *Proceedings of the International Conference on Machine Learning*.
[16] R. Ranganath, D. Tran, D. M. Blei. *arXiv preprint arXiv:1511.02386* **2015**, 1511.02386.
[17] G. Roeder, Y. Wu, D. Duvenaud. *arXiv preprint arXiv:1703.09194* **2017**.
[18] S. Bowman, L. Vilnis, O. Vinyals, A. Dai, R. Jozefowicz, S. Bengio. *arXiv preprint arXiv:1511.06349* **2015**.
[19] R. Gómez-Bombarelli, D. Duvenaud, J. M. Hernndez-Lobato, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams, A. Aspuru-Guzik, *arXiv preprint, arXiv:1610.02415* **2016**, 1610.02415.
[20] J. J. Irwin, T. Sterling, M. M. Mysinger, E. S. Bolstad, R. G. Coleman, *J. Chem. Inf. Model.* **2012**, *52*, 1757–1768.
[21] R. Gómez-Bombarelli, J. Aguilera-Iparraguirre, T. D. Hirzel, D. Duvenaud, D. Maclaurin, M. A. Blood- Forsythe, H. S. Chae, M. Einzinger, D.-G. Ha, T. Wu, G. Markopoulos, S. Jeon, H. Kang, H. Miyazaki, M. Numata, S. Kim, W. Huang, S. I. Hong, M. Baldo, R. P. Adams, A. Aspuru-Guzik, *Nat. Mater.* **2016**, *15*, 1120–1127.
[22] M. J. Kusner, B. Paige, J. M. Hernndez-Lobato, In *International Conference on Machine Learning*. pp. 1945–1954.
[23] P. B. Jørgensen, M. Mesta, S. Shil, J. M. G. Lastra, K. W. Jacobsen, K. S. Thygensen, M. N. Schmidt. Machine learning-based screening of complex molecules for polymer solar cells, **2017**. Unpublished, to be submitted to Journal of Chemical Physics: Special Issue on Data-Enabled Theoretical Chemistry.
[24] D. Kingma, J. Ba, *arXiv preprint arXiv:1412.6980* **2014**. 1412.6980.
[25] E. Brochu, V. M. Cora, N. de Freitas, *arXiv preprint: arXiv:1012.2599* **2010**, 1012.2599.
[26] D. P. Kingma, D. J. Rezende, S. Mohamed, M. Welling, In *Proceedings of the International Conference on Machine Learning*.
[27] L. Rampasek, D. Hidru, P. Smirnov, B. Haibe-Kains, A. Goldenberg, *arXiv preprint, arXiv:1706.08203* **2017**, 1706.08203.
[28] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio. In *Advances in Neural Information Processing Systems.* **2014**.
[29] A. Kadurin, S. Nikolenko, K. Khrabrov, A. Aliper, A. Zhavoronkov, *Mol. Pharm.* **2017**, *14*, 3098–3104.
[30] M. Olivecrona, T. Blaschke, O. Engkvist, H. Chen, *J. Cheminform.* **2017**, *9*, 48.
[31] G. L. Guimaraes, B. Sanchez-Lengeling, P. L. C. Farias, A. Aspuru-Guzik, *arXiv preprint, arXiv:1705.10843* **2017**, *1705*, 10843.

Received: November 3, 2017
Accepted: January 13, 2018
Published online on February 6, 2018

 © 2018 Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim

APPENDIX D

Machine learning-based screening of complex molecules for polymer solar cells

The following paper is published as,

Peter Bjørn Jørgensen, Murat Mesta, Suranjan Shil, Juan Maria García Lastra, Karsten Wedel Jacobsen, Kristian Sommer Thygesen, and Mikkel N. Schmidt. 2018. "Machine Learning-Based Screening of Complex Molecules for Polymer Solar Cells." The Journal of Chemical Physics 148 (24): 241735.

Machine learning-based screening of complex molecules for polymer solar cells

Peter Bjørn Jørgensen,[1,a)] Murat Mesta,[2] Suranjan Shil,[3] Juan Maria García Lastra,[2] Karsten Wedel Jacobsen,[3] Kristian Sommer Thygesen,[3] and Mikkel N. Schmidt[1]
[1]*Department of Applied Mathematics and Computer Science, Technical University of Denmark, Richard Petersens Plads, 2800 Kongens Lyngby, Denmark*
[2]*Department of Energy Conversion and Storage, Technical University of Denmark, Fysikvej, 2800 Kongens Lyngby, Denmark*
[3]*Department of Physics, Technical University of Denmark, Fysikvej, 2800 Kongens Lyngby, Denmark*

(Received 25 January 2018; accepted 9 May 2018; published online 6 June 2018)

Polymer solar cells admit numerous potential advantages including low energy payback time and scalable high-speed manufacturing, but the power conversion efficiency is currently lower than for their inorganic counterparts. In a Phenyl-C_61-Butyric-Acid-Methyl-Ester (PCBM)-based blended polymer solar cell, the optical gap of the polymer and the energetic alignment of the lowest unoccupied molecular orbital (LUMO) of the polymer and the PCBM are crucial for the device efficiency. Searching for new and better materials for polymer solar cells is a computationally costly affair using density functional theory (DFT) calculations. In this work, we propose a screening procedure using a simple string representation for a promising class of donor-acceptor polymers in conjunction with a grammar variational autoencoder. The model is trained on a dataset of 3989 monomers obtained from DFT calculations and is able to predict LUMO and the lowest optical transition energy for unseen molecules with mean absolute errors of 43 and 74 meV, respectively, without knowledge of the atomic positions. We demonstrate the merit of the model for generating new molecules with the desired LUMO and optical gap energies which increases the chance of finding suitable polymers by more than a factor of five in comparison to the randomised search used in gathering the training set. *Published by AIP Publishing.* https://doi.org/10.1063/1.5023563

I. INTRODUCTION

Polymer solar cells have numerous advantages over their inorganic counterparts including mechanical flexibility, lower energy payback time, low or nonexisting ecotoxicity, and fully scalable high-speed manufacturing. However, for organic solar cell technology to become competitive their power conversion efficiency must be increased. So far, the highest reported power conversion efficiency for polymer solar cells are around 10%[1] which is significantly lower than commercial inorganic cells such as silicon, CdS, or CIGS (CuInGaS).

Synthetic chemistry offers a high degree of flexibility to the design polymers with tailored energy levels optimized to the solar spectrum. One particularly promising strategy is to use donor-acceptor (D-A) polymers, which allows for efficient and selective tuning of the highest occupied molecular orbital (HOMO) and lowest unoccupied molecular orbital (LUMO) by varying the D and A units, respectively.[2–7] The HOMO and LUMO energies, as well as other relevant properties, can be determined, at least semi-quantitatively, using *ab inito* calculations such as density functional theory (DFT). However, while such calculations are faster to perform than the experimental synthesis and characterization, the large size and complex structure of D-A polymers relevant to photovoltaics still pose a challenge, and in practice, it is not possible to exhaust the search space by brute force calculations. In practical computational screening, different design choices can be imposed to restrict the search space and the pool of promising candidates is then filtered through one or several low-cost filtering steps. For example, the computational screening of organic photovoltaic materials has been performed using genetic algorithm sampling and PM6 simulations,[8] an artificial neural network trained on existing data to filter out candidate materials based on their fingerprint,[9] and an automatic generation scheme on thiophene-based D-A polymers to evaluate the photovoltaic characteristics with the Hückel models.[10] For small molecules, machine learning algorithms applied to big databases of density functional theory (DFT) calculations have already shown promising results toward achieving DFT-level accuracy at a fraction of the computational cost.[11] Recent work in deep generative models[12,13] has demonstrated the ability to generate new molecular structures based on "simplified molecular-input line-entry system" (SMILES)[14] string representation. This leads us toward a machine learning guided screening strategy in which the model learns to generate and score new D-A polymers. The best suggested candidate polymers can then be assessed with DFT calculations and subsequently synthesized if deemed suitable. In this work, we propose a simplified string representation for donor-acceptor polymers and demonstrate the success of

[a)]Electronic mail: pbjo@dtu.dk

this screening strategy for the search for better polymer solar cells.

The paper is organised as follows: In Sec. II, we first discuss the basic principles of polymer solar cells and identify the key performance limiting parameters used to score the molecules. In Sec. III, we introduce the dataset of polymers and describe three ways of representing the data for the machine learning algorithms. This is followed by Sec. IV with a summary of the recent developments in machine learning methods for prediction of molecular properties. In Sec. V, five different machine learning algorithms are described and each of them is assessed through numerical experiments. Finally, we demonstrate the use of GrammarVAE to guide the DFT-screening effort toward high-performance polymers.

II. BACKGROUND

A. Basics principles of polymer solar cells

In most organic solar cells, the polymer is used as the active photoabsorber. Bound electron-hole pairs, i.e., excitons, created in the polymer diffuse to the acceptor, usually Phenyl-C_61-Butyric-Acid-Methyl-Ester (PCBM), where the exciton dissociates into an electron in the LUMO of the PCBM and a hole in the polymer HOMO/valence band. The dielectric constants of organic polymers are generally quite low ($\epsilon \sim$ 2-4). Consequently, the excitons have rather large binding energies of 0.4–1.4 eV and small excitonic radii[15] so that the electron and hole are mostly localized on the same monomer of the polymer. This implies that the energy of the exciton can be deduced from a calculation of the monomer only—a fact that is exploited in the present work to obtain the onset of absorption (optical gap, ε_{opt}) of the polymer by calculating the energy of an electron-hole pair on a monomer.

To harvest sufficient solar energy, the optical gap of the polymer should be in the range of 1.1-1.7 eV. This condition provides a critical criterion for the design of polymers for solar cells.

Polymers consisting of donor-acceptor (D-A) units have shown particular promise for high-efficiency organic photovoltaics.[16–20] In such polymers, the energy and spatial form of the HOMO is mainly controlled by the donor, while the LUMO is governed by the acceptor unit. The D-A construction entails a natural separation of the electron and the hole thereby minimizing recombination losses. Moreover, tuning of the donor and acceptor units represents an effective way to improve the efficiency of organic solar cells via optimization of the polymer HOMO-LUMO gap (and thus the optical gap).[21]

The correlation between DFT calculated and experimental HOMO and LUMO energies for model D-A compounds has been explored by Blouin and co-workers.[22] This study reveals that the alternating copolymer HOMO energy level is controlled by the electron-donating carbazole moiety, whereas the LUMO energy level mainly depends on the nature of the electron-withdrawing comonomer. The electronic and optical properties of six selected homopolymers have been studied by Zhang *et al.*[23] using two different methods, namely, the oligomer method and DFT with periodic boundary conditions. Using the oligomer method, the HOMO-LUMO gap of the polymer can be estimated by $1/n$ extrapolation of the oligomer gaps. Their calculations show that PBC-DFT is most suited for weakly conjugated homopolymers while the oligomer approach provides a more accurate description for more strongly conjugated polymers. In this work, we focus on the lowest optical transition, i.e., the exciton energy, rather than the HOMO-LUMO gap. The localized nature of the exciton allows us to obtain realistic estimates for the polymer from calculations on a single D-A monomer.

B. Descriptors and machine learning models

In the last few years, several descriptors and models for predicting DFT-calculated properties have emerged. Faber *et al.*[11] benchmark different feature representations and regression algorithms on the QM9[24,25] dataset. Representations include the Coulomb matrix,[26] bag-of-bonds,[27] bonding angular machine learning (BAML),[28] extended-connectivity fingerprints (ECFP),[29] molecular graphs, and the author's distribution based representations such as histograms of distances (HD), angles (HDA,MARAD), and dihedrals (HDAD). The graph convolution on molecular graphs outperforms the other algorithms in terms of accuracy, particularly for the properties that are most important for our application: the HOMO and LUMO energy, HOMO-LUMO gap, and lowest optical transition energy ($\varepsilon_{\text{HOMO}}$, $\varepsilon_{\text{LUMO}}$, ε_{gap}, ε_{opt}). Similar results have been found for other variations of graph convolution/message-passing neural networks on molecular graphs.[30,31]

The feature representations mentioned above are all based on knowing the spatial arrangement of the atoms. The spatial information however is not readily available when searching for new materials and we shift our focus toward algorithms not relying on this information. A generative approach based on variational autoencoders[32,33] is proposed by Gómez-Bombarelli *et al.*[12] The proposed method consist of using the character-based variational autoencoder on SMILES[14] strings to embed the molecules into a continuous vector space. The autoencoder learns a probabilistic encoder and decoder that maps SMILES strings to/from the vector space. The model is trained on 250 000 drug-like molecules from the ZINC database[34] and 100 000 OLED molecules.[35] Then a regression model based on Gaussian processes is trained on the embedding, and it is used to find new proposals for molecules that exhibit the desired property (water-octanol partition coefficient for drug-like molecules). One of the problems with this approach is that the generative model sometimes produces invalid SMILES strings or invalid molecules (with syntactically valid SMILES strings).

The generation of invalid SMILES strings problem has been alleviated by the introduction of the grammar variational autoencoder (GrammarVAE).[13] The SMILES strings can be parsed using a context free grammar, and the parse tree is used as a representation for the molecules. The grammar rules are enforced during decoding, and therefore the generative model can only generate syntactically valid strings. The molecules may still be semantically invalid, e.g., one of the atoms violates valence constraints by having too many

bonds. Avoiding these errors in generative models is an area of active research.[36,37] Arguably the GrammarVAE learns a more coherent latent representation in which nearby points decode to similar strings.[13] We employ the GrammarVAE model on a different context free grammar formulation, tailored to describe the class of molecules we are interested in for designing polymer solar cells. By using the more specific grammar, we further reduce the problem of generating invalid molecules. We describe the GrammarVAE and how we use it in more details in Sec. IV E. Our grammar formulation and SMILES can both be seen as domain specific encodings of molecular graphs. Encoding and generation of general graphs is an avenue of active research with promising recent developments.[38–40] Simonovsky and Komodakis[38] propose a model that generates a probabilistic graph adjacency matrix and node strengths, while You *et al.*[39] and Li *et al.*[40] pose graph generation as a sequence of graph construction operations. Both Simonovsky and Komodakis[38] and Li *et al.*[40] successfully demonstrate generation of small molecules based on the QM9 dataset, but scaling of the models to larger molecules is problematic in both cases.[38,40] The largest molecules of QM9 consist of 29 atoms including hydrogen, while the monomers in our dataset have up to 161 atoms. We therefore approach the problem using simpler encodings of the molecular graphs.

Many researchers are trying to model the HOMO-LUMO gap of organic polymers. Wu *et al.* have proposed the infinite chain descriptor to predict the polymer properties from small units using support vector machine and linear models with feature selection.[41] Pereira *et al.* has benchmarked a range of connectivity-based descriptors for organic molecules for prediction of HOMO and LUMO energies.[42] Their random forest model predicts HOMO and LUMO energies with MAE of 0.15 eV and 0.16 eV, respectively, using PubChem[43] fingerprints. Here we study several machine learning methodologies to predict the DFT-calculated HOMO, LUMO, and the ΔSCF lowest optical transition energy in donor-acceptor type monomers which are described in Sec. III.

III. DATASET OF DONOR-ACCEPTOR POLYMERS

The aim of the present work is to establish a computationally efficient methodology to identify promising D-A molecules for use in polymers solar cells based on the basic electronic and optical properties of the monomers. Quantitative prediction of the performance of a given polymer is an outstanding and presently unsolved challenge that inevitably involves the modeling and coupling of various processes taking place at very different time and length scales, e.g., photoexcitation, energy and charge transport, and charge separation. Rather than venturing into a full-fledged multi-scale description, we follow a different strategy by focusing on a few necessary conditions which can be imposed on the monomers and which can be predicted with high accuracy using *ab initio* calculations.

To this end, we have constructed a computational repository of donor-acceptor monomers whose backbone is generated from a set of 13 acceptor and 10 donor submonomer moieties. Introducing nine possible side-chains and atomic substitutions, more than 10^{14} potential monomer structures can be generated. The donor and acceptor units are shown in Fig. 1.

Although it is required for an efficient chemical synthesis to have as few substitutions as possible, we do not pay

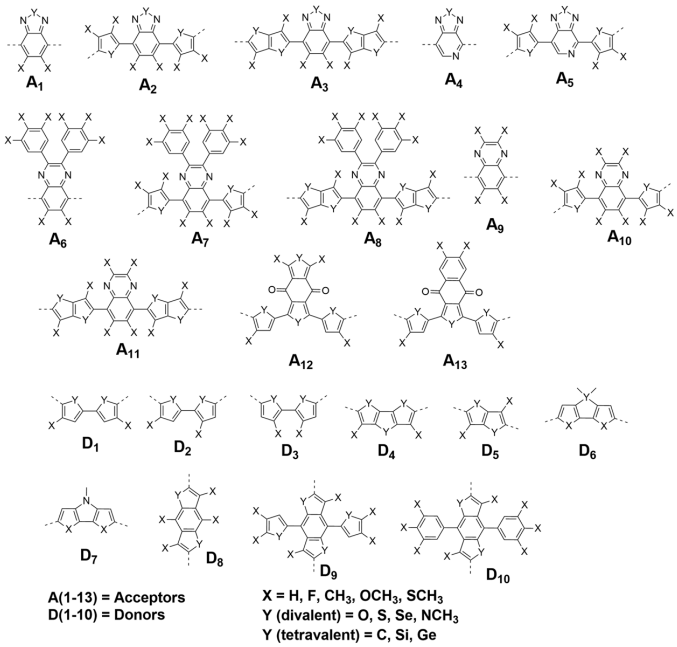


FIG. 1. Donor and acceptor units of the molecules.

attention to the number of different side groups and their relative positions on the backbone when designing the new structures by machine learning; that is, we do not penalize high complexity of the molecules. All the active sites of the monomers are attached to a functional group randomly chosen from a set of possible side groups. Initial structure of the donor-acceptor monomers are generated and pre-optimized using the Merck Molecular Force-Field (MMFF94)[44] within the software Open Babel.[45] We relax the initial geometries further using DFT[46] with the hybrid functional B3LYP[47] and a TZVP basis set[48] all implemented within the Gaussian09 software package.[49]

One way to calculate the lowest optical transition energy is to use the ΔSCF (Δ-Self Consistent Field) method. In this method, the occupation of the molecular orbitals are constrained to simulate the electronic configuration of an electron excited from the HOMO to the LUMO. The optical gap can then be obtained as the difference between the ground state energy and the excited state energy, $\varepsilon_{opt} = E_1 - E_0$. To obtain the lowest optically active transition energy, the excited state should be a singlet, i.e., the total spin of the ground and excited states should be equal. A practical problem with this approach is that the SCF cycle for the excited state configuration can be difficult to converge and often requires tweaking of numerical parameters. This makes it difficult to apply the ΔSCF in a high-throughput context.

A technically simpler approach is to use the lowest triplet excitation energy as an approximation to the singlet. In practice, the triplet is much easier to calculate (the SCF cycle converges faster and is more stable) because it can be obtained by constraining the total spin magnetic moment rather than the individual occupation numbers. Using a test set of 104 monomers, we have found[50] that the difference in energy between the triplet and singlet excited states typically is below 0.3 eV, which is in agreement with results obtained for other organic compounds.[7] In view of the smallness of the energy difference between the triplet and singlet excited states, we have used the triplet excitation energy for the optical gap.

In this work, our primary goal is to distinguish efficient machine-learning methodologies that are relevant for predicting polymer properties. The success of such approaches undoubtedly depends on how extensive the training set is. Our training set consists of high-level DFT calculations of monomer structures spanning a set of promising solar cell polymer materials, which in principle can be combined with the results of dimer, trimer, etc., extrapolating to the polymer. The chain length saturation depends on the system itself particularly the size of the monomer; for instance, in Ref. 51, the saturation length is shown to occur at approximately 6 and 4 repeat units for homo and D-A type conjugated polymers, respectively. In the recent literature, there are several examples of the methods to extrapolate properties from single units to infinite chains, e.g., Refs. 52 and 53; however, we do not apply any kinds of extrapolation in this work because the required oligomer DFT-simulations are highly time consuming for our D-A systems (in a future work, we will address oligomers preferably using less expensive DFT methods). Indeed we expect the machine learning methods that work well for monomers to also be successful when trained on data for polymers. Here the machine learning is performed on a data-set consisting of 3989 monomers split into 5 folds for cross-validation.

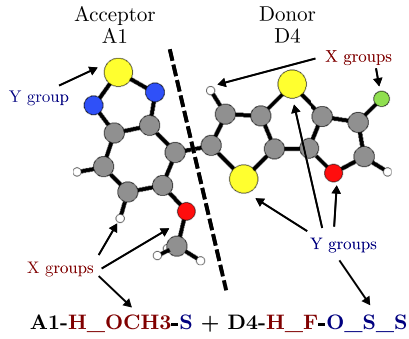


FIG. 2. String representation of one of the molecules of the solar cell dataset: "Acceptor backbone"-"X groups"-"Y groups" + "Donor backbone"-"X groups"-"Y groups." Whenever no side groups are present "*" character is used instead.

A. Feature representation

The monomers defining the polymer structures are composed by a number of smaller building blocks, namely, an acceptor and donor backbone each of which may have none or several X and Y side groups (see Fig. 2). We use three different representations to build our machine learning methods: (a) Fixed length vector representation, (b) string representation using context free grammar, and (c) atomic species and their position (XYZ-coordinates).

1. Fixed length vector representation

The encoding as a fixed length feature vector is described in Table I. The feature vector is expanded by including all products between the one-hot encoded and the count features, yielding a total of 599 features. Notice that in this representation, we simply count the number of times each side group appears as an acceptor/donor X/Y side group in the monomer. Thus the order of the side groups is ignored.

2. String representation

The GrammarVAE[13] is based on encoding context free grammar parse trees **x** to/from a latent vector space **z**. The

TABLE I. Description of features.

Position	Coding	Features
Acceptor	One-hot	A1, A2, A3, A4, A5, A6, A7, A8, A9, A10, A11, A12, A13
Acceptor side groups X	Count	OCH3, CH3, SCH3, F, H
Acceptor side groups Y	Count	S, Se, NCH3, O
Donor	One-hot	D1, D2, D3, D4, D5, D6, D7, D8, D9, D10
Donor side groups X	Count	SCH3, F, H, O, OCH3, S, CH3, Se
Donor side groups Y	Count	S, O, C, Si, Ge, Se, NCH3

original work uses SMILES[14] strings to represent molecules. As we are not interested in the space of all possible molecules, we use a simpler string representation than SMILES. The string representation refers to the building blocks as described above. An example is shown in Fig. 2. The full context free grammar formulation of the string representation is shown in Fig. 9 (Appendix B). The use of a simpler grammar instead of SMILES makes it easier for the GrammarVAE model to encode/decode without discrepancies between the encoded and decoded string. However, it also means that the model is unable to generalise to molecules that do not adhere to the grammar rules. The SMILES grammar is designed to encode any molecule, but this does not necessarily mean that training the GrammarVAE model with SMILES representation will generalise well. As the text representation does not contain any information about the physical properties of different elements, one cannot expect the GrammarVAE model to reliably encode/decode molecular structures that are vastly different from those in the training set.

3. XYZ-coordinates

Note that the two feature descriptors above do not use any spatial information about the molecules and have no information of which atoms constitute an acceptor or donor structure. The third feature representation includes this information, i.e., the atomic species and Cartesian coordinates of each atom of the molecule. The coordinates are obtained through costly DFT calculations, which is the operation we want to avoid in the first place by using machine learning. However, we include this feature to serve as a comparison benchmark and to assess how much the accuracy of the machine learning algorithms can be improved by including the DFT-relaxed molecular geometries as input.

IV. MACHINE LEARNING ALGORITHMS

In the following, we consider several machine learning approaches: As a baseline, we include three classical algorithms: linear ridge regression, multi-layer perceptron, and random forest regression. Then, we move on to discuss two deep neural network approaches, which we consider as the state of the art: The Deep Tensor Neural Network[54] and the Grammar Variational Autoencoder.[13] The data flow for linear ridge regression, multi-layer perceptron, and the deep learning models is illustrated in Fig. 3.

A. Linear model

We employ a linear ridge regression model using the fixed length vector representation of the molecules as input. With linear regression, predictions can be computed as the dot

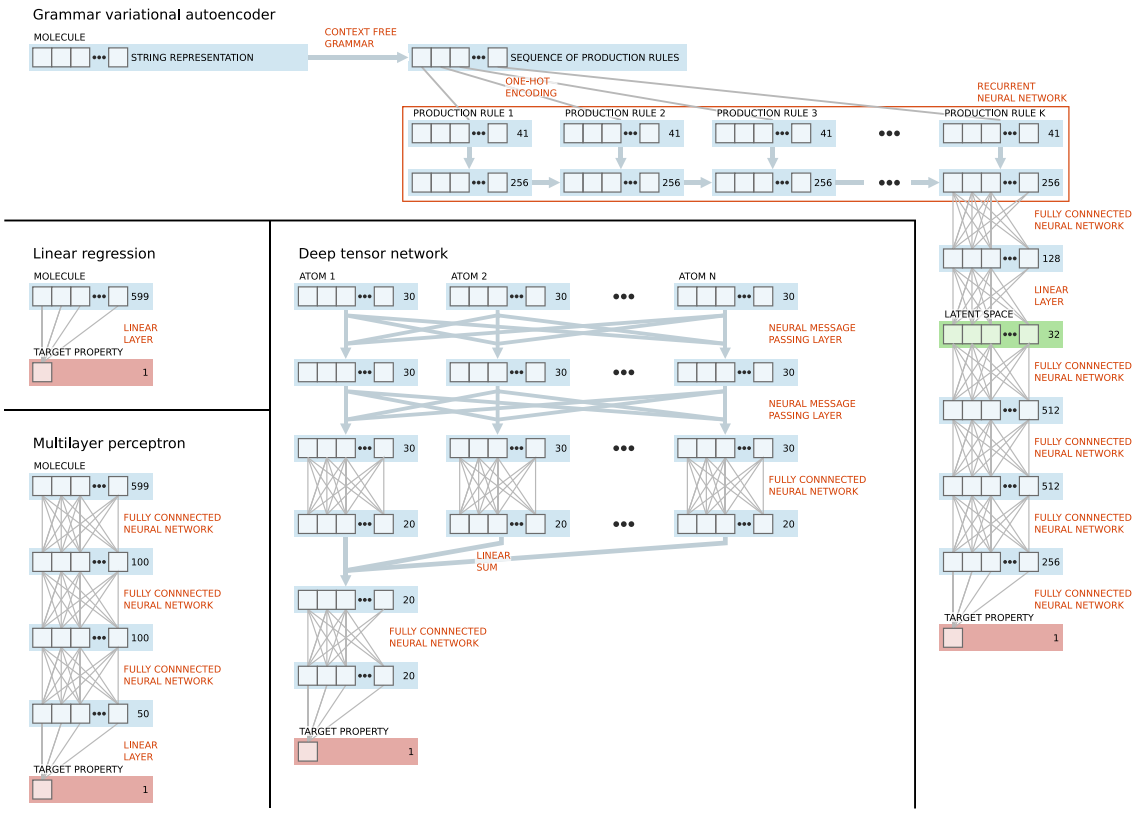


FIG. 3. Data flow of linear regression, multilayer perceptron, deep tensor network, and grammar variational autoencoder.

product of the input $\mathbf{x}$ and the weight vector $\hat{\mathbf{w}}$. The weight vector is determined by minimising the cost function

$$f_{\text{lin}}(\mathbf{w}) = ||\mathbf{X}\mathbf{w} - \mathbf{y}||_2^2 + \beta||\mathbf{w}||_2^2, \quad (1)$$

where $\mathbf{y}$ is a column vector containing the target property of the training data and $\mathbf{X}$ is a matrix where each row corresponds to one training data input and each column is an input feature. We perform the minimisation using Scikit-learn's[55] ridge regression with built-in cross-validation (`RidgeCV`) for the regularisation parameter $\beta \in \left[10^{-4}, 10^{2}\right]$.

B. Multilayer perceptron

Using the fixed length vector representation of the molecules as input, we use a multilayer perceptron, also known as a feedforward neural network. In essence, we replace the linear predictor of the linear ridge regression with a non-linear $g_{\mathbf{w}}(\cdot)$ neural network model, thus the cost function is

$$f_{\text{mlp}}(\mathbf{w}) = \sum_i \left[(g_{\mathbf{w}}(\mathbf{x}_i) - y_i)^2\right] + \beta||\mathbf{w}||_2^2. \quad (2)$$

For $g_{\mathbf{w}}(\cdot)$, we use a 3-hidden-layer (100, 100, 50 hidden units) multilayer perceptron (MLP) neural network with tanh activation functions. Unlike linear ridge regression, there is no closed form solution that minimises the cost function, and therefore the model is trained using Scikit-learn's L-BFGS algorithm with early stopping (using 10 percent of the training data for validation). Again the input feature is the fixed length vector representation. The weight regularisation parameter β is set to 10^{-4}.

C. Random forest

A random forest is an ensemble of decision trees, each trained on a subset of the data, and it uses averaging to make predictions and reduce overfitting. We use the Scikit-learn implementation with default parameters and train the model using the fixed length vector representation as input.

D. Deep tensor neural network

The deep tensor neural network (DTNN)[54] is a deep learning based method that uses the atomic numbers and interatomic distances as input features and maps to a scalar output. The mapping contains two main components, a message passing step followed by a readout step. Each atom is represented as a vector that represents the atom and its local chemical environment. The local environment information is updated by exchanging information with the other atoms, which can be seen as passing messages between the atoms. The readout step maps the set of local environments to a scalar target property. How the atoms interact with each other and the mapping from local environments to the target property is defined by (trainable) neural networks. We expand upon the original architecture[54] to make the model suitable for predicting orbital energies rather than the atomization energy. See Appendix A for a detailed description of the architecture.

TABLE II. Amount of training data used for hyper-parameter optimisation for each model.

Model	Val. set	Hyper-parameters
GrammarVAE	10%	Early stopping $\dim(\mathbf{z}) \in \{16, 32\}$ $\alpha \in \{0.02, 0.04, 0.08\}$ $\lambda \in \{10^{-5}, 10^{-7}\}$ $d_p \in \{0.1, 0.2, 0.3\}$
Linear	10-fold	$\beta \in \{0.0001, 0.000\,72, 0.005\,18, 0.0373, 0.268, 1.93, 13.9, 100\}$
MLP	10%	Early stopping
Random forest	0%	...
Deep tensor	0%	...

TABLE III. Test mean absolute errors (meV) for the models using 5-fold cross-validation.

Model	Input feature	$\varepsilon_{\text{HOMO}}$	$\varepsilon_{\text{LUMO}}$	ε_{gap}	ε_{opt}
GrammarVAE	String	66	43	74	70
Linear	Fixed length	78	51	89	84
MLP	Fixed length	76	46	87	85
Random forest	Fixed length	82	52	94	77
Deep tensor	XYZ	45	31	56	51

E. Grammar variational autoencoder

In an autoencoder, the goal is to learn a representation of the data in terms of an encoder and a decoder which maps

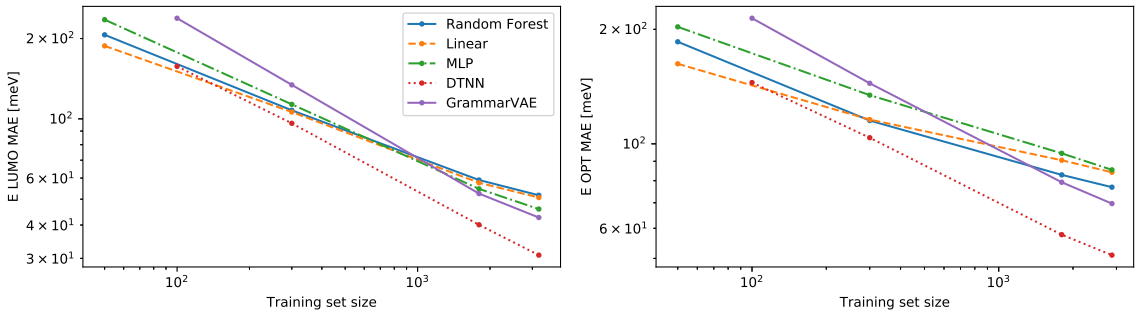


FIG. 4. Mean absolute error in 5-fold cross-validation for different training set sizes.

input data **x** to and from values **z** in a continuous vector space. We use the string representation as input and learn a mapping from this variable-length discrete input to a continuous vector representation. After learning an efficient representation, we can do regression and optimisation in the learned vector space. We use a variational autoencoder (VAE),[32,33] which enables regularisation of the problem by imposing a prior probability density on the distribution of the data in the (latent) vector space. This in turn leads to smoother encoder and decoder functions. We use the GrammarVAE,[13]

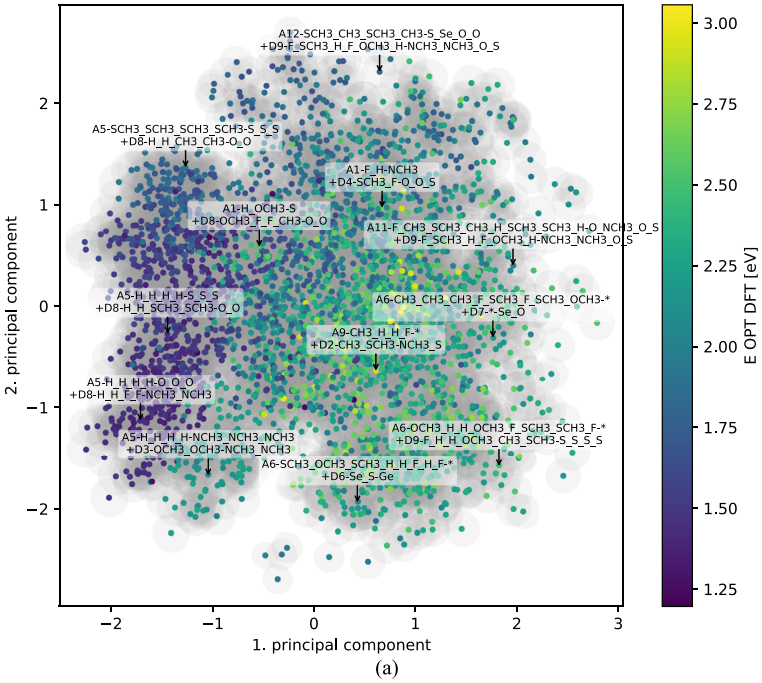


FIG. 5. Projection of the 32-dimensional latent space into the plane defined by the two first principal components. Each point corresponds to the mean value of the embedding distribution $q(\mathbf{z}|\mathbf{x})$, and the gray circles show the contour corresponding to 0.5 standard deviation of the variational distribution. The plot shows all the samples in the data set except those for which the triplet calculations did not converge. (a) The points are coloured according to the DFT-calculated ε_{opt}. (b) The bright points are the ones within the target range for both $\varepsilon_{\text{LUMO}}$ and ε_{opt}.

which extends the VAE such that the decoder always produces syntactically valid output strings. In Appendix B, we describe the architecture and its parameters, and we show an example of encoding and decoding the string representation of a polymer.

V. NUMERICAL RESULTS

A. Predicting molecular properties

For each of the presented models, we want to estimate the generalisation error for predicting $\varepsilon_{\text{HOMO}}$, $\varepsilon_{\text{LUMO}}$, ε_{gap}, and ε_{opt} of unseen molecules. We use 5-fold cross-validation using the data and folds described in Sec. III. For some of the models, we use some of the training data for hyper-parameter selection through another "inner loop" of cross-validation. The fraction of the data used for the hyper-parameter search and the parameters are shown in Table II.

The mean absolute errors (meV) for the regression models are shown in Table III. The Deep Tensor Neural Network outperforms the other models, as could be expected since it is the only model using the spatial information of the molecules. The accuracy of the remaining methods are quite similar, but the GrammarVAE model achieves the lowest MAE across all four properties.

Algorithm 1. Iterated conditional modes.

Input: target $\varepsilon_{\text{LUMO}}$ t_{L}, target ε_{opt} t_{O}
$\varepsilon_{\text{LUMO}}$ regression function f_{L}, ε_{opt} regression function f_{O}
$\hat{\mathbf{z}} \leftarrow \mathcal{N}(\mathbf{0}, \mathbf{I})$
for i = 1 . . . 20 **do**
  $\hat{\mathbf{x}} \leftarrow$ greedy $\underset{\mathbf{x}}{\text{argmax}}\, p_{\boldsymbol{\theta}}(\mathbf{x}|\hat{\mathbf{z}})$
  $\hat{\mathbf{x}} \leftarrow$ RemoveDuplicates($\hat{\mathbf{x}}$)
  $\hat{\mathbf{z}} \leftarrow \underset{\mathbf{z}}{\text{argmax}}\, q_{\boldsymbol{\phi}}(\mathbf{z}|\hat{\mathbf{x}})$
end for
$\mathcal{P} \leftarrow \underset{\{\mathcal{I}\subset\mathbb{N}:|\mathcal{I}|=100\}}{\text{argmin}} \sum_{\mathcal{I}} (t_{\text{L}} - f_{\text{L}}(z_i))^2 + (t_{\text{O}} - f_{\text{O}}(z_i))^2$
return $\hat{\mathbf{x}}_{\mathcal{P}}$

The MAE for $\varepsilon_{\text{LUMO}}$ and ε_{opt} for the different methods versus the training set size is shown in Fig. 4. We notice that the GrammarVAE performs worse than other models for small training set sizes, but its learning curve

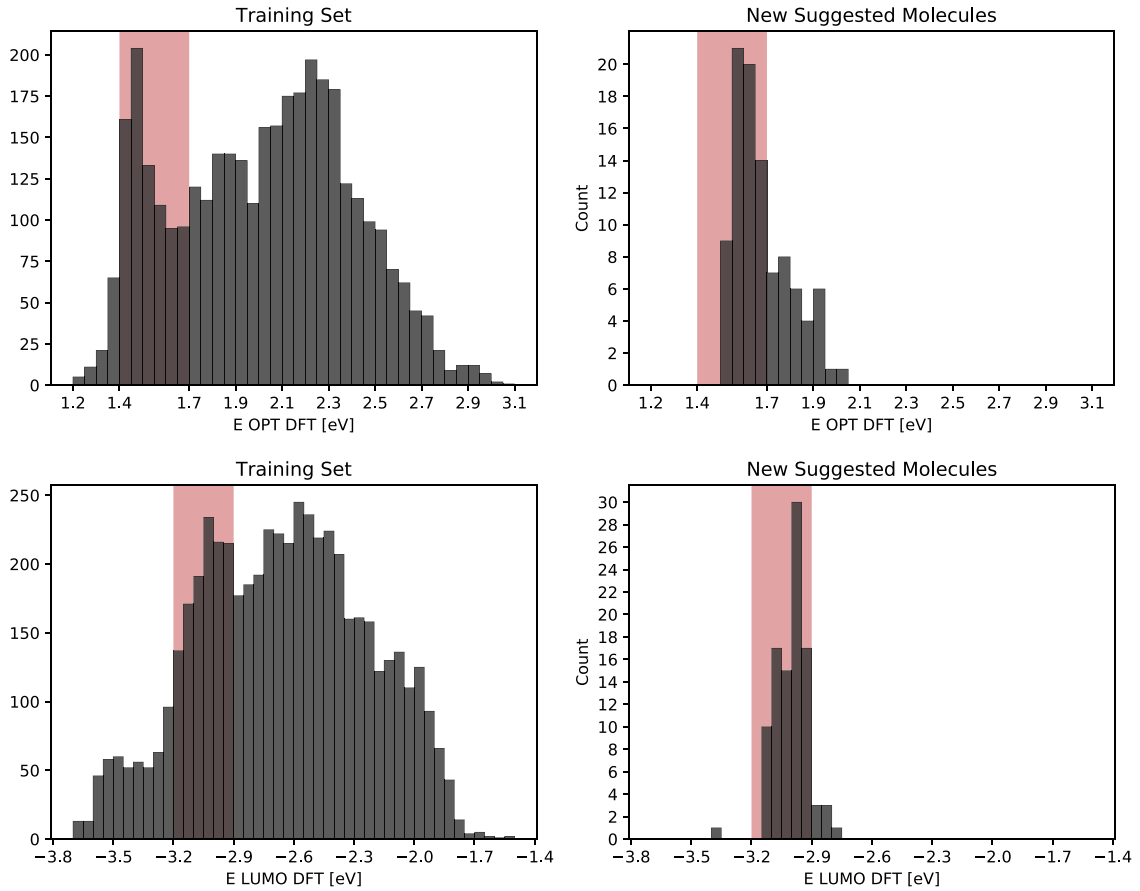


FIG. 6. Histogram for targeted properties (ε_{opt} top, $\varepsilon_{\text{LUMO}}$ bottom) for training set (left) and new suggested molecules (right). The shaded area denotes target range.

is steeper and the other methods, which do not use atomic positions as input, are outperformed when using the full data set.

B. Generating molecules with desired properties

We now test whether we can guide the search for better materials by using the GrammarVAE generative model combined with latent space regression. To generate new candidate molecules, we train a GrammarVAE model using the whole data set as the training set (10% of the training set is still used for hyper-parameter selection).

For each setting of hyper-parameters, we obtain a set of model parameters for the GrammarVAE and the latent space regression model. We select the model for which the average MAE for predicting $\varepsilon_{\mathrm{LUMO}}$ and $\varepsilon_{\mathrm{opt}}$ is minimised on the validation set. The resulting model has a 32-dimensional latent space which is visualized in Fig. 5. Although the embedding has been created using unsupervised model training, it is clear that the model has learned some structure in the data which is correlated with the optical transition energy. This is what can be exploited by the regression model.

Given the latent space embedding and the regression model, the question is, now, how to choose new molecules to use as candidates. In Refs. 12 and 13, a Gaussian process regression model is trained on the latent space representation, and they use Bayesian optimisation to choose new points in the latent space. Bayesian optimisation avoids sampling new points that are close to the training data. As noted in Ref. 12, this may lead to sampling from the latent space where the generative model is not well-behaved. Instead they train a neural network on the latent space. Then they start from the embedding corresponding to a "good" known molecule and take a few gradient steps in the direction that improves the property as predicted by the neural network. We tried this approach but the model was unable to move away from the region corresponding to the "good" training samples. Instead we sample 10^6 samples from the latent space prior $p(\mathbf{z})$ and alternate between decoding using the mode of $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$ [the true mode is approximated by selecting the production rule that maximises (B9) for each time step] and encoding using the mode of $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$. This can be seen as an approximation of iterated conditional modes,[56] where we use the variational approximation $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ instead of the true posterior $p(\mathbf{z}|\mathbf{x})$ to find modes of $p(\mathbf{x}, \mathbf{z})$. After a fixed number of iterations, we evaluate the embedded samples using the neural network regression functions and select the best 100. To speed up the computations, after each iteration we eliminate duplicates (points that decode to the same string) and points that decode to strings that are in the training set. The algorithm is summarised with a pseudo code in Algorithm 1.

We choose the 100 best molecules as predicted by the model and perform DFT calculations on these. Out of the 100 calculations, 3 did not converge. The DFT-calculated $\varepsilon_{\mathrm{opt}}$ and $\varepsilon_{\mathrm{LUMO}}$ of the suggested molecules are shown in Fig. 6. The fraction of the suggested molecules that are within the $\varepsilon_{\mathrm{opt}}$ target is 66% versus 22% for the training set. For the $\varepsilon_{\mathrm{LUMO}}$ target, the fractions are 92% and 22% for the generated molecules and the training set, respectively. The fraction for which both $\varepsilon_{\mathrm{opt}}$ and $\varepsilon_{\mathrm{LUMO}}$ are within the target range is 61% for the suggested molecules and 11% for the training set.

VI. CONCLUSION

Discovery of new materials for polymer solar cells might enable cheaper sustainable energy solutions in the future. Random screening with density functional theory calculations alone is computationally costly. We have demonstrated the use of machine learning algorithms to predict the lowest unoccupied molecular orbital (LUMO) energy and the lowest optical transition energy calculated using the triplet ΔSCF method. All the considered models achieve mean absolute prediction errors for the LUMO and gap below 52 meV and 85 meV, respectively. Using a Grammar Variational Autoencoder with a simple application specific context free grammar, we create a vector space embedding of the data points from which new molecules can be generated. By using a neural network regression model, we select only molecules that are likely to have properties within the target ranges. This guided search yields a success rate of 61% versus 11% for the random exploration employed in generating the training set.

Our variant of the Deep Tensor Neural Network shows higher predictive accuracy than doing predictions based on the Grammar Variational Autoencoder embedding, but it uses the atomic positions as inputs. The atomic positions are calculated using DFT calculations, and the Deep Tensor Neural Network is therefore not immediately applicable to new unseen molecules. Future work should look into whether we can achieve accurate predictions using atomic positions approximations or perhaps the positions can also be estimated with machine learning.

It is important to note that the LUMO position and the optical absorption edge are only two among several parameters that determine the efficiency of a particular polymer in an organic photovoltaic cell. Particularly influential factors include charge transport within and between polymer chains or the electron-hole recombination probability. We saw that only 11% of the polymers in the random poll studied here would be suitable polymers for solar cells according to their LUMO position and optical edge. This percentage would drastically decrease with the inclusion of other requirements, such as the ones mentioned above. Furthermore, charge mobility and electron-hole recombination probability are quantities which are much more demanding to calculate than the LUMO or the optical edge. This points toward the enormous potential that machine learning has in this area, both for guiding the search of suitable candidates and for avoiding prohibitive computations. Further work along these lines is now in progress.

ACKNOWLEDGMENTS

The authors thank the Innovation Fund Denmark [INKA-Inks for large-scale processing of polymer solar cells (Grant No. 5160-0008B)]. Support from VILLUM FONDEN (No. 9455) is gratefully acknowledged. This research has been supported by the NVIDIA Corporation with a donation of one Titan X GPU.

APPENDIX A: DEEP TENSOR NEURAL NETWORK IMPLEMENTATION

The Deep Tensor Neural Network (DTNN) model can be interpreted as a message passing algorithm operating on a molecular graph.[31] Each atomic species has its own embedding of trainable parameters $\mathbf{c}_a \in \mathbb{R}^{30}$, and each atom is represented by its embedding, which corresponds to the node features of the graph. We draw an edge between all pairs of nodes in the graph. The edge feature is a radial basis function expansion of the distance between two atoms. Denoting the distance between atom i and j as $d_{i,j}$, the edge between node i and j in the graph has the feature vector

$$(\mathbf{r}_{i,j})_k = \exp\left(-\frac{(d_{i,j} - (-\mu_{\min} + k\Delta))^2}{2\Delta^2}\right), \tag{A1}$$

where for our application we have chosen $\mu_{\min} = -1$, $\Delta = 0.2$, $k = 0\ldots49$. In each step of the message-passing algorithm, the nodes are updated in parallel by summing over all the incoming messages,

$$\mathbf{c}_i^{(t+1)} = \mathbf{c}_i^{(t)} + \sum_{j \neq i} m(\mathbf{c}_j^{(t)}, \mathbf{r}_{i,j}). \tag{A2}$$

The message function is given by

$$m(\mathbf{c}_j, \mathbf{r}_{i,j}) = \tanh\left[\mathbf{W}^{fc}((\mathbf{W}^{cf}\mathbf{c}_j + \mathbf{b}^{f_1}) \circ (\mathbf{W}^{df}\mathbf{r}_{i,j} + \mathbf{b}^{f_2}))\right], \tag{A3}$$

where $\circ$ denotes element-wise multiplication and $\left\{\mathbf{W}^{fc} \in \mathbb{R}^{30\times60}, \mathbf{W}^{cf} \in \mathbb{R}^{60\times30}, \mathbf{W}^{df} \in \mathbb{R}^{60\times50}\right\}$ and $\left\{\mathbf{b}^{f_1} \in \mathbb{R}^{60}, \mathbf{b}^{f_2} \in \mathbb{R}^{60}\right\}$ are trainable parameters of the model. After a number of these interaction steps (we use 2), the nodes are combined into a single entity for the entire graph. We do this with a per node fully connected layer after which all the nodes are summed, followed by three fully connected layers. The network structure is shown in Fig. 7. The model is implemented in Theano[57] and trained using stochastic gradient descent with a squared error cost function.

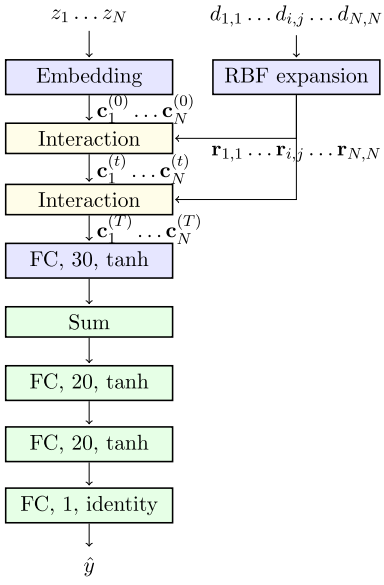


FIG. 7. Flow of (forward-pass) computations in our variation of the Deep Tensor Neural Network (DTNN). The operations marked with "FC" are fully connected neural network layers where the number denotes the number of hidden units followed by the name of the activation function used.

APPENDIX B: GRAMMARVAE IMPLEMENTATION

We want to learn an encoder and a decoder which maps input data $\mathbf{x}$ to and from values $\mathbf{z}$ in a continuous vector space. In variational autoencoders,[32,33] $\mathbf{z}$ is treated as a latent variable of a generative model. The likelihood function $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$ parameterised by $\boldsymbol{\theta}$ can be interpreted as a probabilistic decoder, which maps a latent vector representation $\mathbf{z}$ into a probability distribution over data $\mathbf{x}$. To train the model, we seek to maximise the model (log) evidence,

$$\log p(\mathbf{x}) = \log \mathbb{E}_{p(\mathbf{z})}\left[p_\theta(\mathbf{x}|\mathbf{z})\right]. \tag{B1}$$

Computing the expectation is intractable, so we instead turn to a cost function based on the evidence lower bound

$$\log p(\mathbf{x}) \geq \ell(\mathbf{x}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) - \alpha \log \frac{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}{p(\mathbf{z})}\right], \tag{B2}$$

where $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ is the variational approximation of the posterior $p(\mathbf{z}|\mathbf{x})$ with parameters $\boldsymbol{\phi}$ which constitutes a probabilistic encoder. The parameterised functions $\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$ and $\log q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ are implemented as deep neural networks. The evidence lower bound is maximised using stochastic gradient ascent with respect to $\{\boldsymbol{\theta}, \boldsymbol{\phi}\}$ where the expectation in (B2) is approximated using a few samples from $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$. We have introduced a temperature parameter $\alpha \in [0{:}1]$, which is set to 1 when evaluating the lower bound but is adjusted during training to trade-off between the reconstruction term $\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})\right]$ and the encoder complexity penalty term $\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[-\log \frac{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}{p(\mathbf{z})}\right] = -\mathrm{KL}\left[q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\right]$, where $\mathrm{KL}[q||p]$ denotes the Kullback-Leibler divergence from p to q. Consistent with other variational autoencoder applications,[58–60] we also found that without the temperature parameter the encoding function $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ would converge to the prior $p(\mathbf{z})$ yielding poor reconstruction performance. Others have handled this issue by annealing the α parameter during training[59,60] or by employing a different objective function,[58] which effectively ignores the KL-term when it is below a certain threshold. In this work, we keep the α parameter fixed during stochastic gradient descent training but select the value using cross-validation.

The GrammarVAE encodes to and decodes from the latent space as illustrated in Fig. 8; see the original paper[13] for a detailed description. In contrast to the original GrammarVAE,[13] we employ "teacher forcing" for the decoding recurrent neural network, i.e., the sampled (output) production rule at time n is fed as input to the recurrent neural network at time step $n + 1$. We found that this reduces the number of required training steps to reach convergence significantly. To reduce notational clutter, in the following $\mathbf{x}$ refers to the sequence of 1-hot encoded production rules and we use $\mathbf{x}^{(t)}$ to refer to the tth production rule of the sequence.

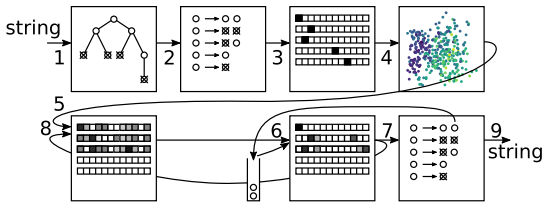


FIG. 8. The grammar variational autoencoder maps strings to/from vector space in a number of steps. (1) Form parse tree. (2) Convert tree to a sequence of production rules. (3) Convert rules into 1-hot encoded vectors. (4) Embed sequence into vector space using recurrent neural network. (5) Output logit vector ($p(\mathbf{x}^{(1)}|\mathbf{z})$) conditioned on embedding using recurrent neural network. (6) Mask invalid production rules conditioned on non-terminal popped from top of the stack. If stack is empty, go to (9). (7) Sample production rule and push non-terminals to stack. (8) Output logit vector ($p(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)}\cdots\mathbf{x}^{(1)},\mathbf{z})$) conditioned on embedding and sampled production rules using recurrent neural network and go to step (6). (9) Convert sampled production rules to molecule description string.

As in the original GrammarVAE,[13] we choose a fully factorised normal distribution as the variational distribution,

$$q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) = \mathcal{N}\big(\mathbf{z}|\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x}), \boldsymbol{\sigma}^2_{\boldsymbol{\phi}}(\mathbf{x})\big), \tag{B3}$$

where the mean and (log)variance are parameterised by a neural network consisting of 256 gated recurrent units (GRU),[61] followed by a 128-unit fully connected layer with a linear rectifier activation function and separate linear layers for the mean and log-variance, i.e.,

$$\mathbf{d}(\mathbf{y}) = \texttt{Dense}(\texttt{GRU}(\mathbf{y})), \tag{B4}$$

$$\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{y}) = \texttt{Linear}\,(\mathbf{d}(\mathbf{y})), \tag{B5}$$

$$\boldsymbol{\sigma}^2_{\boldsymbol{\theta}}(\mathbf{y}) = \exp\left(\texttt{Linear}\,(\mathbf{d}(\mathbf{y}))\right). \tag{B6}$$

Similarly the probabilistic decoder model can be written as

$$p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{0},\mathbf{I}), \tag{B7}$$

$$p_{\boldsymbol{\theta}}(\mathbf{x}^{(1)}|\mathbf{z}) = \mathcal{C}\left(\mathbf{x}^{(1)}|\boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{z})\right), \tag{B8}$$

$$p_{\boldsymbol{\theta}}(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)}\cdots\mathbf{x}^{(1)},\mathbf{z}) = \mathcal{C}\left(\mathbf{x}^{(t)}|\boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{z},\mathbf{x}^{(t-1)}\cdots\mathbf{x}^{(1)})\right), \tag{B9}$$

where $\mathcal{C}(\cdot|\boldsymbol{\pi})$ is the categorical distribution with the parameter $\boldsymbol{\pi}$. The function $\boldsymbol{\pi}(\cdot)$ assigns probability to each production rule and is implemented by a 256-unit GRU followed by a linear layer unit, which is then normalised to produce a probability distribution by a softmax layer, i.e.,

$$\boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{z},\mathbf{x}^{(t-1)}\cdots\mathbf{x}^{(1)}) = \texttt{softmax}\left(\texttt{Linear}\left(\texttt{GRU}(\mathbf{z},\mathbf{x}^{(t-1)}\cdots\mathbf{x}^{(1)})\right)\right). \tag{B10}$$

```
  m → A - OPTG - OPTG + D - OPTG - OPTG
  A → A1 | A2 | A3 | A4 | A5 | A6 | A7 | A8 | A9 | A10 | A11 | A12 | A13
  D → D1 | D2 | D3 | D4 | D5 | D6 | D7 | D8 | D9 | D10
OPTG → * | GS
 GS → G | GS _ G
  G → Ge | CH3 | OCH3 | H | C | O | SCH3 | NCH3 | S | F | Si | Se
```

FIG. 9. Production rules for the context free grammar of donor-acceptor polymers.

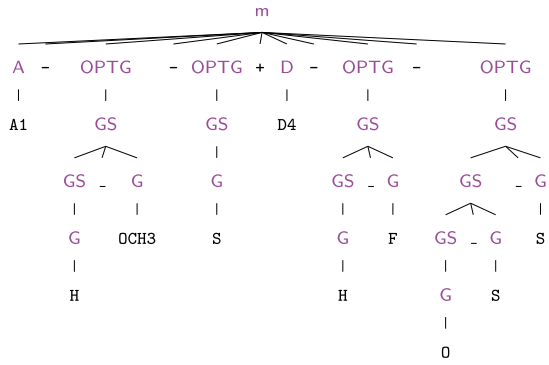


FIG. 10. Parse tree for A1-H_OCH3-S+D4-H_F-O_S_S.

During testing, the GrammarVAE masks and renormalises (B10) for each step such that no probability mass is assigned to invalid production rules.

Once the GrammarVAE model has been trained, we can use it to map the molecular string representation to the latent vector space. We then perform regression in the latent space using a 512-512-256-1 MLP with the tanh activation function. The model is trained using stochastic gradient descent,

```
  m → A - OPTG - OPTG + D - OPTG - OPTG
  A → A1
OPTG → GS
 GS → GS _ G
 GS → G
  G → H
  G → OCH3
OPTG → GS
 GS → G
  G → S
  D → D4
OPTG → GS
 GS → GS _ G
 GS → G
  G → H
  G → F
OPTG → GS
 GS → GS _ G
 GS → GS _ G
 GS → G
  G → O
  G → S
  G → S
```

FIG. 11. Sequence of production rules for A1-H_OCH3-S+D4-H_F-O_S_S.

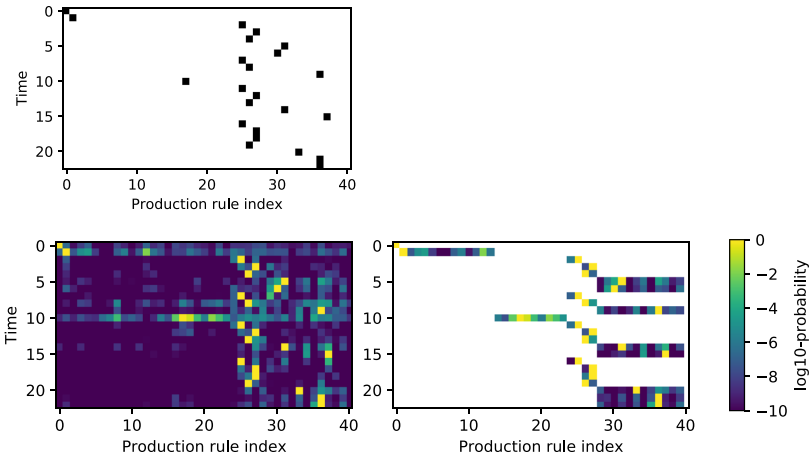


FIG. 12. Top: sequence of production rules encoded into the binary matrix. Bottom left: log-probabilities assigned to production rules after each time step of the decoder. Bottom right: log-probabilities assigned to production rules after applying the mask.

and in each epoch we obtain a new sample for each training data point using the probabilistic encoder $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$. In addition to the regularisation achieved through this sampling scheme, we also add dropout[62] with dropout probability d_p after the second and third hidden layers as well as ℓ_2-regularisation of the weights with regularisation parameter λ. We train a separate MLP regression model for each property of interest, but the same embedding is used for all the properties.

The context free grammer used in our GrammarVAE is defined by the production rules shown in Fig. 9. The coloured symbols denote non-terminals, i.e., symbols that are replaced using the rules of the grammar. There are 41 production rules including a NULL-rule that is used for padding sequences of production rules to the same length.

In the following, we go through the example of encoding the string `A1-H_OCH3-S+D4-H_F-O_S_S` to a vector $\mathbf{z}$ and decoding it back to the original string. The parse tree for the string is shown in Fig. 10. The tree is converted into a sequence by traversing the tree depth first left-to-right, yielding the sequence of production rules shown in Fig. 11. The 23 production rules are 1-hot-encoded into a binary matrix shown in Fig. 12 (top). The recurrent neural network reads the matrix one row at a time, and when it reaches the end of the sequence its 256-dimensional state vector is mapped to the mean and log-variance, from which we can sample using $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) = \mathcal{N}\left(\mathbf{z}|\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x}), \sigma^2_{\boldsymbol{\phi}}(\mathbf{x})\right)$. This gives a 32-dimensional vector $\mathbf{z}$ which is fed as input to the decoding recurrent neural network. The neural network outputs a vector of probabilities for the first production rule, which corresponds to the first row of the bottom left matrix in Fig. 12. After applying the mask, only one production rule is valid, and production rule 0 is therefore sampled with probability 1. The non-terminals of the right hand side of the production rule are pushed to the stack from right to left such that A is now on top of the stack. In the next time step, the recurrent neural network gets production rule 0 as well as $\mathbf{z}$ as input and outputs another vector of probabilities (second row of the bottom left matrix in Fig. 12). We take the first item from the stack (A) and therefore mask out all the production rules that do not have A on the left-hand side. We thus sample A → `A1` with high probability. The production rule is used as input to the recurrent neural network in the next time step and the procedure continues until the stack is empty. In this example, we successfully restore the original sequence of production rules and the decoder thus reproduces the original input string from the embedding $\mathbf{z}$.

[1] M. A. Green, Y. Hishikawa, W. Warta, E. D. Dunlop, D. H. Levi, J. Hohl-Ebinger, and A. W. Ho-Baillie, Prog. Photovoltaics: Res. Appl. **25**, 668 (2017).
[2] M. Turbiez, P. Frère, M. Allain, C. Videlot, J. Ackermann, and J. Roncali, Chem.–Eur. J. **11**, 3742 (2005).
[3] M. Cheng, X. Yang, F. Zhang, J. Zhao, and L. Sun, J. Phys. Chem. C **117**, 9076 (2013).
[4] D. P. Hagberg, T. Marinado, K. M. Karlsson, K. Nonomura, P. Qin, G. Boschloo, T. Brinck, A. Hagfeldt, and L. Sun, J. Org. Chem. **72**, 9550 (2007).
[5] T. Xu and L. Yu, Mater. Today **17**, 11 (2014).
[6] B.-G. Kim, X. Ma, C. Chen, Y. Ie, E. W. Coir, H. Hashemi, Y. Aso, P. F. Green, J. Kieffer, and J. Kim, Adv. Funct. Mater. **23**, 439 (2013).
[7] K. B. Ornso, J. M. Garcia-Lastra, and K. S. Thygesen, Phys. Chem. Chem. Phys. **15**, 19478 (2013).
[8] I. Y. Kanal, S. G. Owens, J. S. Bechtel, and G. R. Hutchison, J. Phys. Chem. Lett. **4**, 1613 (2013).
[9] E. O. Pyzer-Knapp, K. Li, and A. Aspuru-Guzik, Adv. Funct. Mater. **25**, 6495 (2015).
[10] Y. Imamura, M. Tashiro, M. Katouda, and M. Hada, J. Phys. Chem. C **121**, 28275 (2017).
[11] F. A. Faber, L. Hutchison, B. Huang, J. Gilmer, S. S. Schoenholz, G. E. Dahl, O. Vinyals, S. Kearnes, P. F. Riley, and O. A. von Lilienfeld, J. Chem. Theory Comput. **13**, 5255 (2017).
[12] R. Gómez-Bombarelli, D. Duvenaud, J. M. Hernández-Lobato, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams, and A. Aspuru-Guzik, ACS. Cent. Sci. **4**, 268 (2018).
[13] M. J. Kusner, B. Paige, and J. M. Hernández-Lobato, in *Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 6-11 August 2017*, edited by D. Precup and Y. W. Teh (PMLR, 2017), Vol. 70, pp. 1945–1954.
[14] D. Weininger, J. Chem. Inf. Model. **28**, 31 (1988).
[15] I. Hill, A. Kahn, Z. Soos, and J. R. A. Pascal, Chem. Phys. Lett. **327**, 181 (2000).
[16] R. S. Kularatne, H. D. Magurudeniya, P. Sista, M. C. Biewer, and M. C. Stefan, J. Polym. Sci., Part A: Polym. Chem. **51**, 743 (2013).
[17] A. Facchetti, Mater. Today **16**, 123 (2013).

[18]T. E. Kang, J. Choi, H.-H. Cho, S. C. Yoon, and B. J. Kim, Macromolecules **49**, 2096 (2016).
[19]Z. Li, Y. Zang, C.-C. Chueh, N. Cho, J. Lu, X. Wang, J. Huang, C.-Z. Li, J. Yu, and A. K.-Y. Jen, Macromolecules **47**, 7407 (2014).
[20]E. Bundgaard and F. C. Krebs, "Low band gap polymer materials for organic solar cells," Sol. Energy Mater. Sol. Cells **91**, 954 (2007).
[21]L. Zhang, K. Pei, M. Yu, Y. Huang, H. Zhao, M. Zeng, Y. Wang, and J. Gao, J. Phys. Chem. C **116**, 26154 (2012).
[22]N. Blouin, A. Michaud, D. Gendron, S. Wakim, E. Blair, R. Neagu-Plesu, M. Belletête, G. Durocher, Y. Tao, and M. Leclerc, J. Am. Chem. Soc. **130**, 732 (2008).
[23]L. Zhang, M. Yu, H. Zhao, Y. Wang, and J. Gao, Chem. Phys. Lett. **570**, 153 (2013).
[24]R. Ramakrishnan, P. O. Dral, M. Rupp, and O. A. von Lilienfeld, Sci. Data **1**, 140022 (2014).
[25]L. Ruddigkeit, R. van Deursen, L. C. Blum, and J.-L. Reymond, J. Chem. Inf. Model. **52**, 2864 (2012).
[26]M. Rupp, A. Tkatchenko, K.-R. Müller, and O. A. von Lilienfeld, Phys. Rev. Lett. **108**, 058301 (2012).
[27]K. Hansen, F. Biegler, R. Ramakrishnan, W. Pronobis, O. A. von Lilienfeld, K.-R. Müller, and A. Tkatchenko, J. Phys. Chem. Lett. **6**, 2326 (2015).
[28]B. Huang and O. A. von Lilienfeld, J. Chem. Phys. **145**, 161102 (2016).
[29]D. Rogers and M. Hahn, J. Chem. Inf. Model. **50**, 742 (2010).
[30]K. T. Schütt, P.-J. Kindermans, H. E. Sauceda, S. Chmiela, A. Tkatchenko, and K.-R. Müller, *Advances in Neural Information Processing Systems 30*, edited by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Curran Associates, Inc., 2017), pp. 991–1001.
[31]J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, in *Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 6-11 August 2017*, edited by D. Precup and Y. W. Teh (PMLR, 2017), Vol. 70, pp. 1263–1272.
[32]D. P. Kingma and M. Welling, e-print arXiv:1312.6114v10 [stat.ML] (2013).
[33]D. J. Rezende, S. Mohamed, and D. Wierstra, in *Proceedings of the 31st International Conference on International Conference on Machine Learning* (PMLR, 2014), pp. 1278–1286.
[34]J. J. Irwin, T. Sterling, M. M. Mysinger, E. S. Bolstad, and R. G. Coleman, J. Chem. Inf. Model. **52**, 1757 (2012).
[35]R. Gómez-Bombarelli, J. Aguilera-Iparraguirre, T. D. Hirzel, D. Duvenaud, D. Maclaurin, M. A. Blood-Forsythe, H. S. Chae, M. Einzinger, D.-G. Ha, T. Wu, G. Markopoulos, S. Jeon, H. Kang, H. Miyazaki, M. Numata, S. Kim, W. Huang, S. I. Hong, M. Baldo, R. P. Adams, and A. Aspuru-Guzik, Nat. Mater. **15**, 1120 (2016).
[36]D. Janz, J. van der Westhuizen, B. Paige, M. J. Kusner, and J. M. Hernandez-Labato, e-print arXiv:1712.01664 [stat.ML] (2017).
[37]H. Dai, Y. Tian, B. Dai, S. Skiena, and L. Song, in NIPS 2017 Workshop Machine Learning for Molecules and Materials, 2017.
[38]M. Simonovsky and N. Komodakis, e-print arXiv:1802.03480 [cs.LG] (2018).
[39]J. You, R. Ying, X. Ren, W. L. Hamilton, and J. Leskovec, e-print arXiv:1802.08773 [cs.LG] (2018).
[40]Y. Li, O. Vinyals, C. Dyer, R. Pascanu, and P. Battaglia, e-print arXiv:1803.03324 [cs.LG] (2018).
[41]K. Wu, N. Sukumar, N. A. Lanzillo, C. Wang, R. "Rampi" Ramprasad, R. Ma, A. F. Baldwin, G. Sotzing, and C. Breneman, J. Polym. Sci., Part B: Polym. Phys. **54**, 2082 (2016).
[42]F. Pereira, K. Xiao, D. A. R. S. Latino, C. Wu, Q. Zhang, and J. Aires-de Sousa, J. Chem. Inf. Model. **57**, 11 (2017).
[43]See ftp://ftp.ncbi.nlm.nih.gov/pubchem/specifications/pubchem_fingerprints.txt for Pubchem substructure fingerprint; accessed 21 November 2017.
[44]T. A. Halgren, J. Comput. Chem. **17**, 616 (1996).
[45]N. M. O'Boyle, M. Banck, C. A. James, C. Morley, T. Vandermeersch, and G. R. Hutchison, J. Cheminf. **3**, 33 (2011).
[46]W. Kohn and L. J. Sham, Phys. Rev. **140**, A1133 (1965).
[47]A. D. Becke, J. Chem. Phys. **98**, 5648 (1993).
[48]A. Schäfer, C. Huber, and R. Ahlrichs, J. Chem. Phys. **100**, 5829 (1994).
[49]M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, G. Scalmani, V. Barone, G. A. Petersson, H. Nakatsuji, X. Li, M. Caricato, A. Marenich, J. Bloino, B. G. Janesko, R. Gomperts, B. Mennucci, H. P. Hratchian, J. V. Ortiz, A. F. Izmaylov, J. L. Sonnenberg, D. Williams-Young, F. Ding, F. Lipparini, F. Egidi, J. Goings, B. Peng, A. Petrone, T. Henderson, D. Ranasinghe, V. G. Zakrzewski, J. Gao, N. Rega, G. Zheng, W. Liang, M. Hada, M. Ehara, K. Toyota, R. Fukuda, J. Hasegawa, M. Ishida, T. Nakajima, Y. Honda, O. Kitao, H. Nakai, T. Vreven, K. Throssell, J. A. Montgomery, Jr., J. E. Peralta, F. Ogliaro, M. Bearpark, J. J. Heyd, E. Brothers, K. N. Kudin, V. N. Staroverov, T. Keith, R. Kobayashi, J. Normand, K. Raghavachari, A. Rendell, J. C. Burant, S. S. Iyengar, J. Tomasi, M. Cossi, J. M. Millam, M. Klene, C. Adamo, R. Cammi, J. W. Ochterski, R. L. Martin, K. Morokuma, O. Farkas, J. B. Foresman, and D. J. Fox, GAUSSIAN 09, Revision E.01, Gaussian, Inc., Wallingford, CT, 2009.
[50]M. Mesta, S. Suranjan, K. S. Thygesen, and J. M. Garcia-Lastra, "First-principles analysis of electronic and optical properties of donor-acceptor type polymer photovoltaic materials" (unpublished).
[51]T. M. McCormick, C. R. Bridges, E. I. Carrera, P. M. DiCarmine, G. L. Gibson, J. Hollinger, L. M. Kozycz, and D. S. Seferos, Macromolecules **46**, 3879 (2013).
[52]J. Torras, J. Casanovas, and C. Alemán, J. Phys. Chem. A **116**, 7571 (2012).
[53]R. E. Larsen, J. Phys. Chem. C **120**, 9650 (2016).
[54]K. T. Schütt, F. Arbabzadah, S. Chmiela, K. R. Müller, and A. Tkatchenko, Nat. Commun. **8**, 13890 (2017).
[55]F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, J. Mach. Learn. Res. **12**, 2825 (2011).
[56]J. Besag, J. R. Stat. Soc.: Ser. B: Methodol. **B-48**, 259 (1986).
[57]Theano Development Team, e-print arXiv:1605.02688 [cs.SC] (2016).
[58]D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling, e-print arXiv:1606.04934 [cs.LG] (2016).
[59]C. Sønderby, T. Raiko, L. Maaløe, S. Sønderby, and O. Winther, "How to train deep variational autoencoders and probabilistic ladder networks," in Proceedings of the 33rd International Conference on Machine Learning (ICML 2016), 2016.
[60]S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Jozefowicz, and S. Bengio, e-print arXiv:1511.06349 [cs.LG] (2015).
[61]K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (Association for Computational Linguistics, 2014), pp. 1724–1734.
[62]N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, J. Mach. Learn. Res. **15**, 1929 (2014).

{2.7182818284}ϑφερτυθιοπσδφγηξκλ