

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

Peter Bjørn Jørgensen,<sup>1, a)</sup> Murat Mesta,<sup>2</sup> Suranjan Shil,<sup>3</sup> Juan Maria García Lastra,<sup>2</sup>  
 Karsten Wedel Jacobsen,<sup>3</sup> Kristian Sommer Thygesen,<sup>3</sup> and Mikkel N. Schmidt<sup>1</sup>

<sup>1)</sup>*Department of Applied Mathematics and Computer Science,  
 Technical University of Denmark*

<sup>2)</sup>*Department of Energy Conversion and Storage, Technical University of Denmark*

<sup>3)</sup>*Department of Physics, Technical University of Denmark*

(Dated: 12 September 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 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.

PACS numbers: Valid PACS appear here

---

<sup>a)</sup>Electronic mail: pbjo@dtu.dk

## I. INTRODUCTION

Polymer solar cells have numerous advantages over their inorganic counterparts including mechanical flexibility, lower energy payback time, low or nonexistent 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%<sup>1</sup> 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.<sup>2-7</sup> The HOMO and LUMO energies, as well as other relevant properties, can be determined, at least semi-quantitatively, using ab-initio 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 have been performed by, using genetic algorithm sampling and PM6 simulations,<sup>8</sup> using artificial neural network trained on existing data to filter out candidate materials based on their fingerprint,<sup>9</sup> using an automatic generation-scheme on thiophene-based D-A polymers to evaluate the photovoltaic characteristics with the Hückel models.<sup>10</sup> For small molecules, machine learning algorithms applied to big databases of density functional theory (DFT) calculations, have already shown promising results towards achieving DFT-level accuracy at a fraction of the computational cost.<sup>11</sup> Recent work in deep generative models<sup>12,13</sup> has demonstrated the ability to generate new molecular structures based on “simplified molecular-input line-entry system” (SMILES)<sup>14</sup> string representation. This leads us towards 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 this screening strategy for the search for better polymer solar cells.

The paper is organised as follows: In section II we first discuss the basic principles of polymer solar cells and identify the key performance limiting parameters used to score the molecules. In section III we introduce the dataset of polymers and describe three ways of representing the data for the machine learning algorithms. This is followed by section IV with a summary of the recent developments in machine learning methods for prediction of molecular properties. In section V, five different machine learning algorithms are described and each of them are assessed through numerical experiments. Finally, we demonstrate the use of GrammarVAE to guide the DFT-screening effort towards 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<sub>61</sub>-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<sup>15</sup> 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,  $\epsilon_{\text{opt}}$ ) 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.<sup>16–20</sup> 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).<sup>21</sup>

The correlation between DFT calculated and experimental HOMO and LUMO energies for model D-A compounds has been explored by Blouin and coworkers.<sup>22</sup> 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.<sup>23</sup> 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.*<sup>11</sup> benchmark different feature representations and regression algorithms on the QM9<sup>24,25</sup> dataset. Representations include the Coulomb matrix<sup>26</sup>, bag-of-bonds<sup>27</sup>, bonding angular machine learning (BAML)<sup>28</sup>, extended-connectivity fingerprints (ECFP)<sup>29</sup>, molecular graphs and the author’s distribution based representations such as histogram 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 ( $\epsilon_{\text{HOMO}}$ ,  $\epsilon_{\text{LUMO}}$ ,  $\epsilon_{\text{gap}}$ ,  $\epsilon_{\text{opt}}$ ). Similar results have been found for other variations of graph convolution/message-passing neural networks on molecular graphs.<sup>30,31</sup>

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 towards algorithms not relying on this information. A generative approach based on variational autoencoders<sup>32,33</sup> is proposed by Gómez-Bombarelli *et al.*<sup>12</sup> The proposed method consist of using character-based variational autoencoder on SMILES<sup>14</sup> 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<sup>34</sup> and 100.000 OLED molecules<sup>35</sup>. Then a regression model based on Gaussian processes is trained on the embedding and it is used to find new proposals for molecules that exhibits 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)<sup>13</sup>. 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.<sup>36,37</sup> Arguably the GrammarVAE learns a more coherent latent representation in which nearby points decode to similar strings.<sup>13</sup> 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 section 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.<sup>38-40</sup> Simonovsky and Komodakis<sup>38</sup> propose a model that generates a probabilistic graph adjacency matrix and node strengths while You *et al.*<sup>39</sup> and Li *et al.*<sup>40</sup> pose graph generation as a sequence of graph construction operations. Both Simonovsky and Komodakis<sup>38</sup> and Li *et al.*<sup>40</sup> successfully demonstrate generation of small molecules based on the QM9 dataset, but scaling of the models to larger

molecules is problematic in both cases.<sup>38,40</sup> 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.<sup>41</sup> Pereira *et al.* has benchmarked a range of connectivity-based descriptors for organic molecules for prediction of HOMO and LUMO energies.<sup>42</sup> Their Random forest model predicts HOMO and LUMO energies with MAE of 0.15 eV and 0.16 eV respectively using PubChem<sup>43</sup> fingerprints. Here we study several machine learning methodologies to predict the DFT-calculated HOMO, LUMO and the  $\Delta$ SCF lowest optical transition energy in donor-acceptor type monomers as described in the following section.

### 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, 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 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

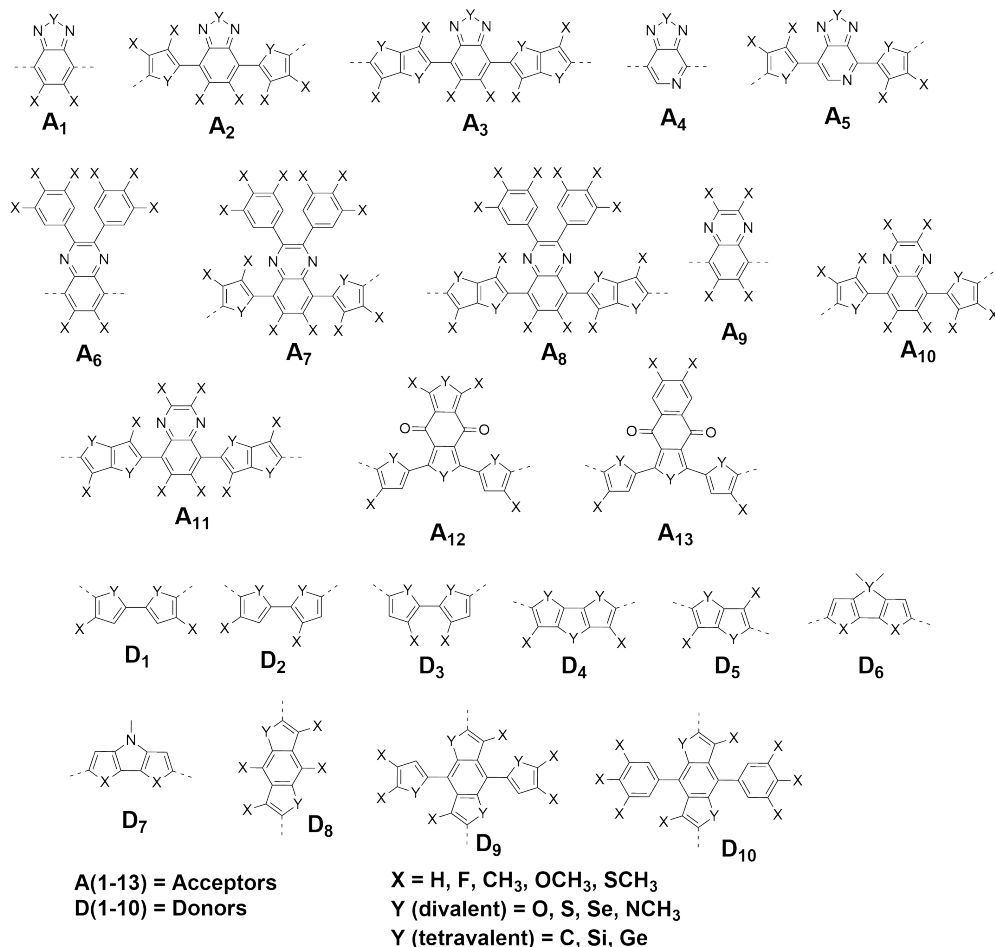


FIG. 1: Donor and acceptor units of the molecules.

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 Merck Molecular Force-Field (MMFF94)<sup>44</sup> within the software Open Babel<sup>45</sup>. We relax the initial geometries further using DFT<sup>46</sup> with the hybrid functional B3LYP<sup>47</sup> and a TZVP basis set<sup>48</sup> all implemented within the Gaussian09 software package<sup>49</sup>.

One way to calculate the lowest optical transition energy is to use the  $\Delta$ SCF ( $\Delta$ -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  $\Delta$ SCF 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<sup>50</sup> 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<sup>7</sup>. 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 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. ref 52,53, 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 that the successful machine-learning methods trained on monomer data to be also valid on polymers as well. Here the machine learning is performed on a data-set consisting of 3989 monomers split into 5 folds for cross-validation.

## 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



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

TABLE I: Description of features

machine learning methods: (a) Fixed length vector representation, (b) string representation using context free grammar, (c) atomic species and their position (XYZ-coordinates).

*a. 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 acceptor/donor X/Y side group in the monomer. Thus the order of the side groups is ignored.

*b. String representation* The GrammarVAE<sup>13</sup> is based on encoding context free grammar parse trees  $\mathbf{x}$  to/from a latent vector space  $\mathbf{z}$ . The original work uses SMILES<sup>14</sup> 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

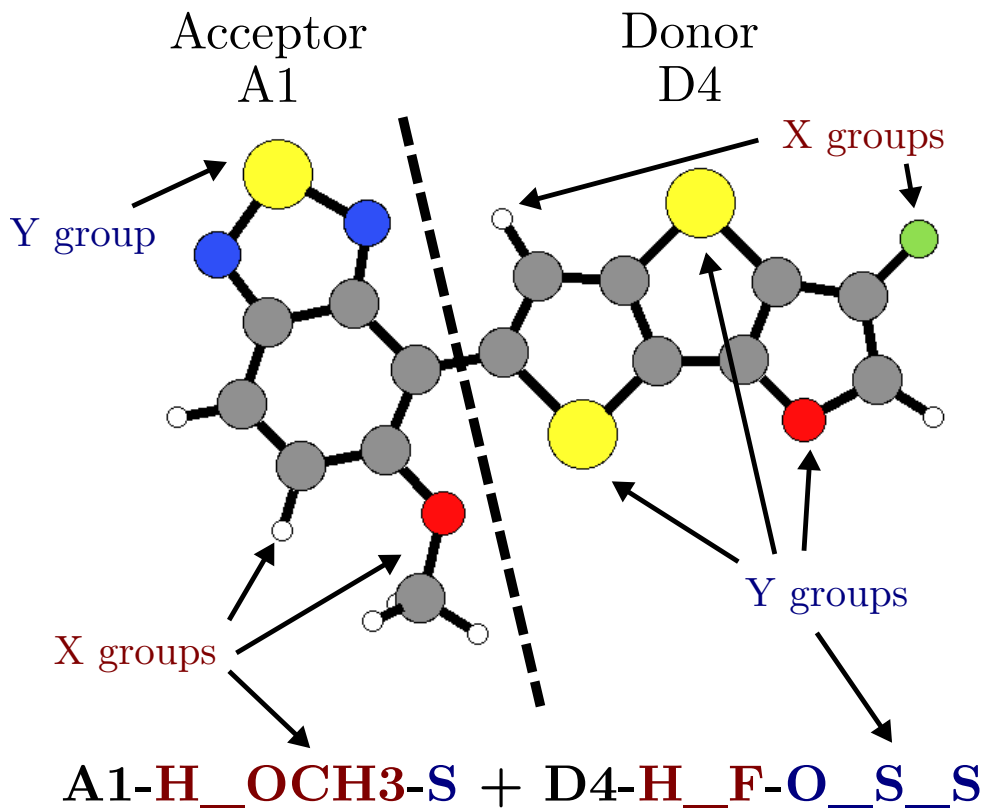


FIG. 2: String representation of one of the molecules of the solar cell dataset: “Acceptor backbone”-“X groups”-“Y groups”+“Donor backbone”-“X groups”-“Y groups”. Whenever no side groups are present “\*” character is used instead.

to reliably encode/decode molecular structures that are vastly different from those in the training set.

*c. 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 coordinate of each atom of the molecule. The coordinates are obtained through costly DFT calculations, the operation we would like to avoid by using machine learning in the first place. However, we include this feature to serve as a comparison benchmark, to assess how much the accuracy of the machine learning algorithms can be improved by including the DFT-relaxed molecular geometries as input.

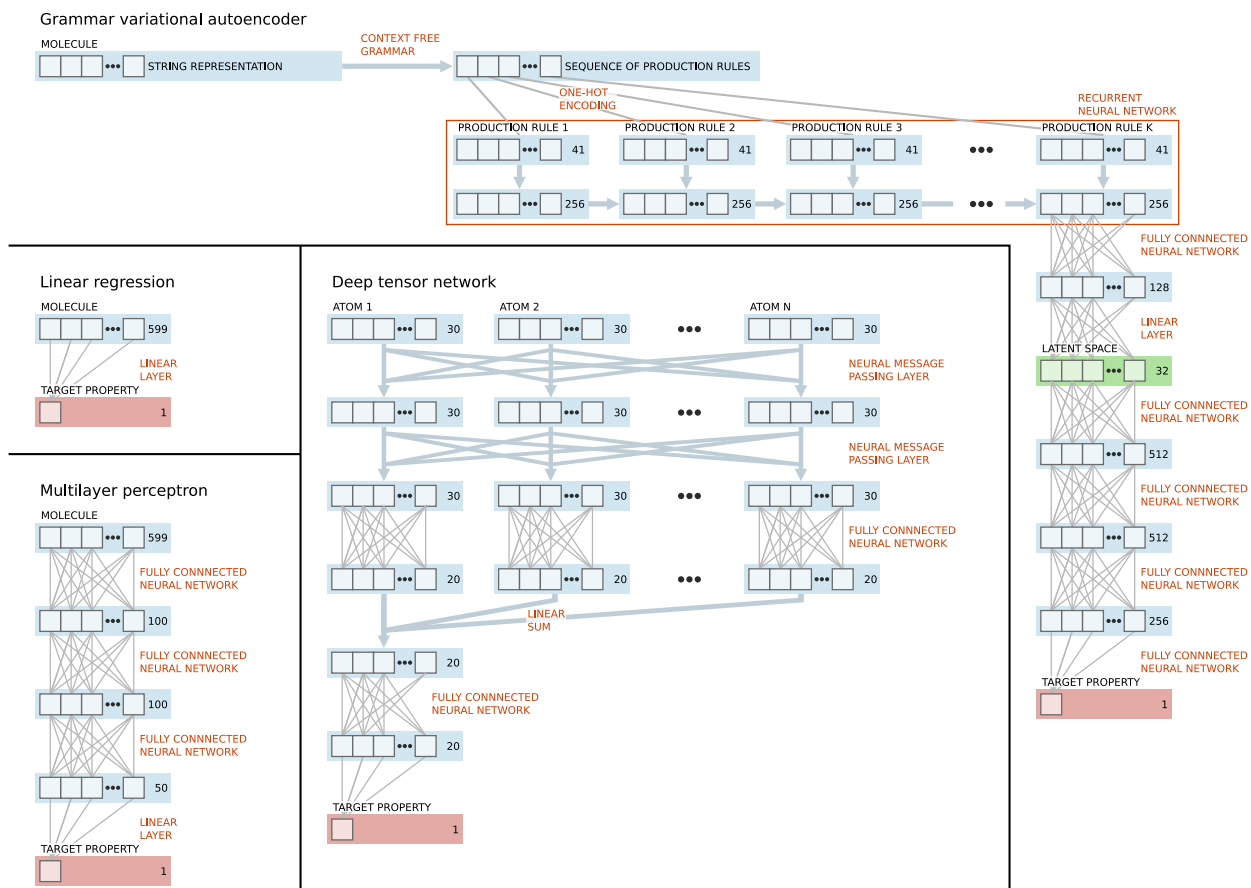


FIG. 3: Data flow of linear regression, multilayer perceptron, deep tensor network, and grammar variational autoencoder.

## 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 the state of the art: The Deep Tensor Neural Network<sup>54</sup> and the Grammar Variational Autoencoder<sup>13</sup>. 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 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<sup>55</sup> ridge regression with built-in cross-validation (**RidgeCV**) for the regularisation parameter  $\beta \in [10^{-4}, 10^2]$ .

## 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

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

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  $\beta$  is set to  $10^{-4}$ .

## C. Random Forest

The random forest algorithm is also using the fixed length vector input. 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.

## D. Deep Tensor Neural Network

The deep tensor neural network (DTNN)<sup>54</sup> 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<sup>54</sup> to make the model suitable for predicting orbital energies rather than the atomization energy. See Appendix A for a detailed description of the architecture.

## 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 input data  $\mathbf{x}$  to and from values  $\mathbf{z}$  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)<sup>32,33</sup>, 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<sup>13</sup>, which extends the VAE such that the decoder always produce syntactically valid output strings. In Appendix B we describe the architecture, its parameters and show an example of encoding and decoding the string representation of a polymer.

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	$\beta \in \{0.0001, 0.00072, 0.00518,$ $0.0373, 0.268, 1.93, 13.9, 100\}$
MLP	10%	Early stopping
Random Forest	0%	—
Deep Tensor	0%	—

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

## V. NUMERICAL RESULTS

### A. Predicting Molecular Properties

For each of the presented models we want to estimate the generalisation error for predicting  $\varepsilon_{\text{HOMO}}$ ,  $\varepsilon_{\text{LUMO}}$ ,  $\varepsilon_{\text{gap}}$  and  $\varepsilon_{\text{opt}}$  of unseen molecules. We use 5-fold cross-validation using the data and folds described in section 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 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.

The MAE for  $\varepsilon_{\text{LUMO}}$  and  $\varepsilon_{\text{opt}}$  for the different methods versus 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 is steeper and the other methods, which do not use

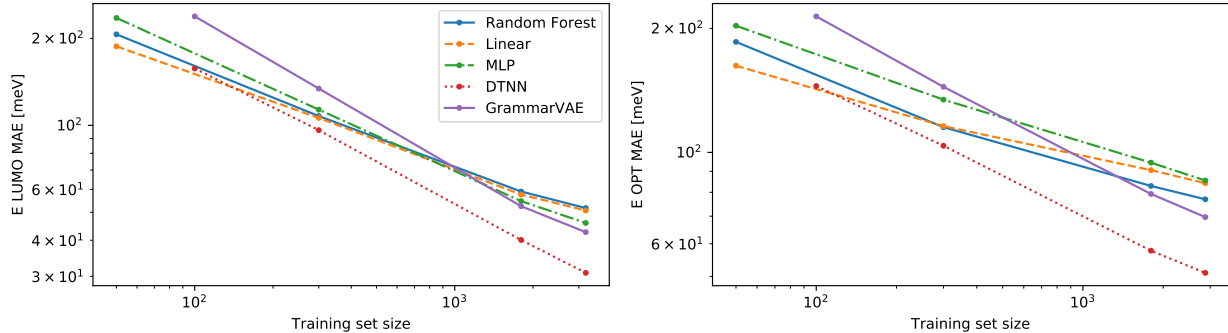


FIG. 4: Mean absolute error in 5-fold cross-validation for different training set sizes

Model	Input Feature	$\varepsilon_{\text{HOMO}}$	$\varepsilon_{\text{LUMO}}$	$\varepsilon_{\text{gap}}$	$\varepsilon_{\text{opt}}$
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

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

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 hyperparameter selection).

For each setting of hyperparameters 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  $\varepsilon_{\text{LUMO}}$  and  $\varepsilon_{\text{opt}}$  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 ref. 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  $p_{\theta}(\mathbf{x}|\mathbf{z})$  (the true mode is approximated by selecting the production rule that maximises (B9) for each time step) and encoding using the mode of  $q_{\phi}(\mathbf{z}|\mathbf{x})$ . This can be seen as an approximation of iterated conditional modes<sup>56</sup>, where we use the variational approximation  $q_{\phi}(\mathbf{z}|\mathbf{x})$  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 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  $\varepsilon_{\text{opt}}$  and  $\varepsilon_{\text{LUMO}}$  of the suggested molecules are shown in FIG. 6. The fraction of the suggested molecules that are within the  $\varepsilon_{\text{opt}}$  target is 66% versus 22% for the training set. For the  $\varepsilon_{\text{LUMO}}$  target the fractions are 92% and 22% for the generated molecules and the training set respectively. The fraction for which both  $\varepsilon_{\text{opt}}$  and  $\varepsilon_{\text{LUMO}}$  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



---

**Algorithm 1** Iterated Conditional Modes

---

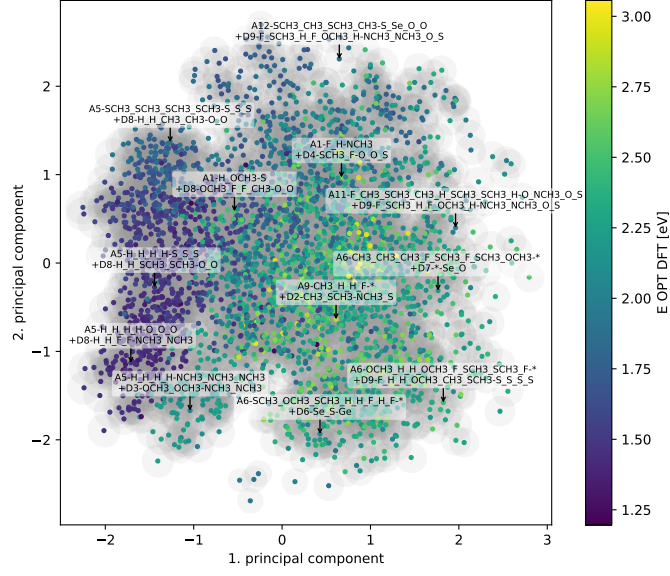
Input: target  $\varepsilon_{\text{LUMO}}$   $t_{\text{L}}$ , target  $\varepsilon_{\text{opt}}$   $t_{\text{O}}$   
 $\varepsilon_{\text{LUMO}}$  regression function  $f_{\text{L}}$ ,  $\varepsilon_{\text{opt}}$  regression function  $f_{\text{O}}$   
 $\hat{\mathbf{z}} \leftarrow \mathcal{N}(\mathbf{0}, \mathbf{I})$   
**for**  $i=1 \dots 20$  **do**  
     $\hat{\mathbf{x}} \leftarrow \underset{\mathbf{x}}{\text{greedy argmax}} p_{\theta}(\mathbf{x}|\hat{\mathbf{z}})$   
     $\hat{\mathbf{x}} \leftarrow \text{RemoveDuplicates}(\hat{\mathbf{x}})$   
     $\hat{\mathbf{z}} \leftarrow \underset{\mathbf{z}}{\text{argmax}} q_{\phi}(\mathbf{z}|\hat{\mathbf{x}})$   
**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**  $\hat{\mathbf{x}}_{\mathcal{P}}$

---

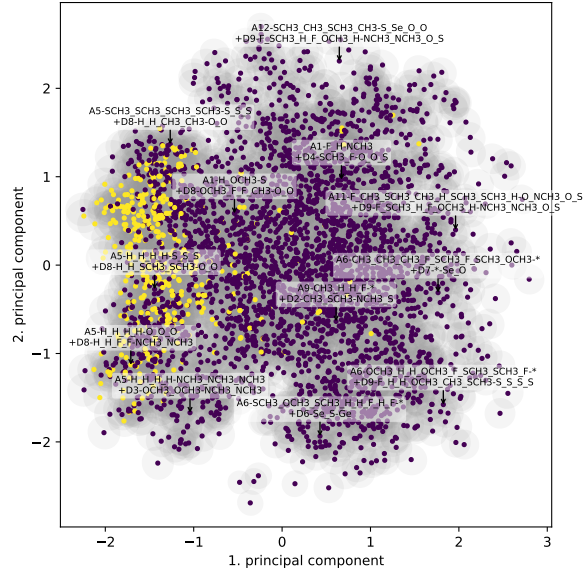
is computationally costly. We have demonstrated the use of machine learning algorithms to predict the lowest unoccupied molecular orbital (LUMO) energy and the lowest optical transition energy calculated using the triplet  $\Delta\text{SCF}$  method. All the considered models achieve mean absolute prediction 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



(a) The points are coloured according to the DFT-calculated  $\epsilon_{\text{opt}}$ .



(b) The bright points are the ones within the target range for both  $\epsilon_{\text{LUMO}}$  and  $\epsilon_{\text{opt}}$ .

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 grey circles shows 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.

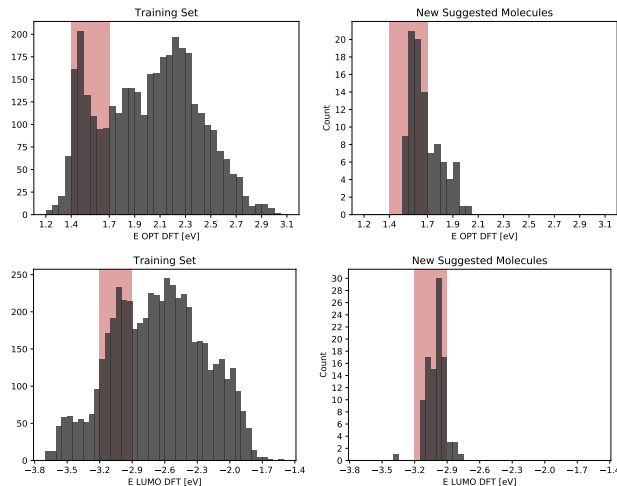


FIG. 6: Histogram for targeted properties ( $\epsilon_{\text{opt}}$  top,  $\epsilon_{\text{LUMO}}$  bottom) for training set (left) and new suggested molecules (right). The shaded area denotes target range.

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 much more demanding to calculate than the LUMO or the optical edge. This points towards 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 undergoing.

## ACKNOWLEDGMENTS

The authors thank the Innovation Fund Denmark (INKA -Inks for large-scale processing of polymer solar cells). Support from VILLUM FONDEN (#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.<sup>31</sup> 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 feature vector

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

where for our application we have chosen  $\mu_{\min} = -1, \Delta = 0.2, k = 0 \dots 49$ . In each step of the message-passing algorithm the nodes are updated in parallel by summing over all the incoming messages

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

The message-function is given by

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

where  $\circ$  denotes element-wise multiplication and  $\{\mathbf{W}^{fc} \in \mathbb{R}^{30 \times 60}, \mathbf{W}^{cf} \in \mathbb{R}^{60 \times 30}, \mathbf{W}^{df} \in \mathbb{R}^{60 \times 50}\}$  and  $\{\mathbf{b}^{f_1} \in \mathbb{R}^{60}, \mathbf{b}^{f_2} \in \mathbb{R}^{60}\}$  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<sup>57</sup> and trained using stochastic gradient descent with a squared error cost function.

## 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<sup>32,33</sup>  $\mathbf{z}$  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

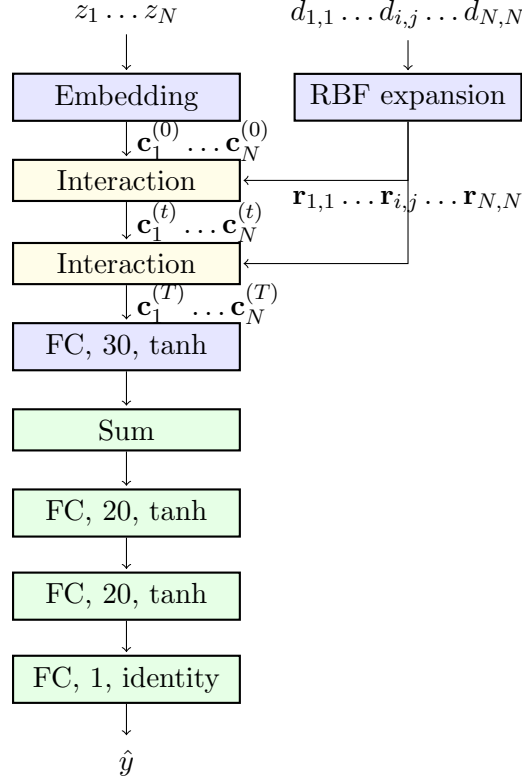


FIG. 7: Flow of (forward-pass) computations in our variation of Deep Tensor Neural Network (DTNN). The operations marked with “FC” are fully connected neural network layers where the number denotes the number of hidden units followed by the name of the activation function used.

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,

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

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

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

where  $q_{\Phi}(\mathbf{z}|\mathbf{x})$  is the variational approximation of the posterior  $p(\mathbf{z}|\mathbf{x})$  with parameters  $\Phi$  which constitutes a probabilistic encoder. The parameterised functions  $\log p_{\theta}(\mathbf{x}|\mathbf{z})$  and  $\log q_{\Phi}(\mathbf{z}|\mathbf{x})$  are implemented as deep neural networks. The evidence lower bound is maximised using stochastic gradient ascent with respect to  $\{\theta, \Phi\}$  where the expectation in (B2) is

approximated using a few samples from  $q_{\Phi}(\mathbf{z}|\mathbf{x})$ . 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  $\mathbb{E}_{q_{\Phi}(\mathbf{z}|\mathbf{x})} [\log p_{\theta}(\mathbf{x}|\mathbf{z})]$  and the encoder complexity penalty term  $\mathbb{E}_{q_{\Phi}(\mathbf{z}|\mathbf{x})} \left[ -\log \frac{q_{\Phi}(\mathbf{z}|\mathbf{x})}{p(\mathbf{z})} \right] = -\text{KL} [q_{\Phi}(\mathbf{z}|\mathbf{x})||p(\mathbf{z})]$ , where  $\text{KL} [q||p]$  denotes the Kullback-Leibler divergence from  $p$  to  $q$ . Consistent with other variational autoencoder applications<sup>58–60</sup> we also found that without the temperature parameter the encoding function  $q_{\Phi}(\mathbf{z}|\mathbf{x})$  would converge to the prior  $p(\mathbf{z})$  yielding poor reconstruction performance. Others have handled this issue by annealing the  $\alpha$  parameter during training<sup>59,60</sup> or by employing a different objective function<sup>58</sup>, which effectively ignores the KL-term when it is below a certain threshold. In this work we keep the  $\alpha$  parameter fixed during stochastic gradient descent training, but select the value using cross-validation.

The grammar VAE encodes to and decodes from the latent space as illustrated in FIG. 8. See the original paper<sup>13</sup> for a detailed description. In contrast to the original grammarVAE<sup>13</sup> 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  $t$ ’th production rule of the sequence.

As in the original Grammar VAE<sup>13</sup> we choose a fully factorised normal distribution as variational distribution:

$$q_{\Phi}(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_{\Phi}(\mathbf{x}), \boldsymbol{\sigma}_{\Phi}^2(\mathbf{x})), \quad (\text{B3})$$

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

$$\mathbf{d}(\mathbf{y}) = \text{Dense}(\text{GRU}(\mathbf{y})), \quad (\text{B4})$$

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

$$\boldsymbol{\sigma}_{\theta}^2(\mathbf{y}) = \exp(\text{Linear}(\mathbf{d}(\mathbf{y}))). \quad (\text{B6})$$

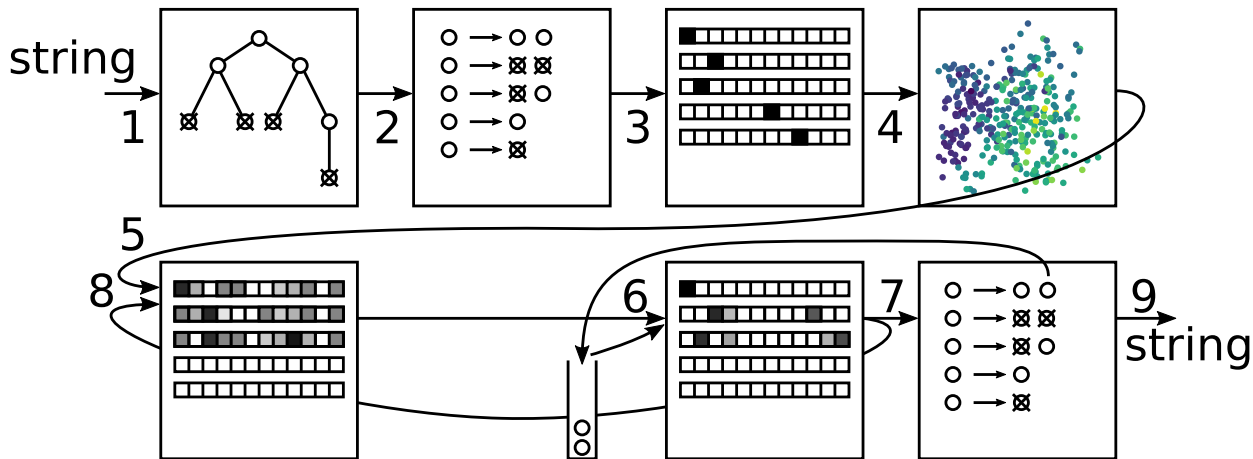


FIG. 8: The grammar variational autoencoder maps strings to/from vector space in a number of steps. (1) Form parse tree. (2) Convert tree to a sequence of production rules. (3) Convert rules into 1-hot encoded vectors. (4) Embed sequence into vector space using recurrent neural network. (5) Output logit 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, goto (9). (7) Sample production rule and push non-terminals to stack. (8) Output logit vector ( $p(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)} \dots \mathbf{x}^{(1)}, \mathbf{z})$ ) 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.

Similarly the probabilistic decoder model can be written as:

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

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

$$p_{\theta}(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)} \dots \mathbf{x}^{(1)}, \mathbf{z}) = \mathcal{C}(\mathbf{x}^{(t)}|\boldsymbol{\pi}_{\theta}(\mathbf{z}, \mathbf{x}^{(t-1)} \dots \mathbf{x}^{(1)})) \quad (\text{B9})$$

where  $\mathcal{C}(\cdot|\boldsymbol{\pi})$  is the categorical distribution with 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.

$$\boldsymbol{\pi}_{\theta}(\mathbf{z}, \mathbf{x}^{(t-1)} \dots \mathbf{x}^{(1)}) = \text{softmax}(\text{Linear}(\text{GRU}(\mathbf{z}, \mathbf{x}^{(t-1)} \dots \mathbf{x}^{(1)}))) \quad (\text{B10})$$

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 tanh activation function. The model is trained using stochastic gradient descent and in each epoch we obtain a new sample for each training data point using the probabilistic encoder  $q_{\Phi}(\mathbf{z}|\mathbf{x})$ . In addition to the regularisation achieved through this sampling scheme we also add dropout<sup>62</sup> with dropout probability  $d_p$  after the second and third hidden layers as well as  $\ell_2$ -regularisation of the weights with regularisation parameter  $\lambda$ . 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 grammar used in our Grammar VAE 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  $q_{\Phi}(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_{\Phi}(\mathbf{x}), \boldsymbol{\sigma}_{\Phi}^2(\mathbf{x}))$ . 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`  $\rightarrow$  `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 reproduce



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

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

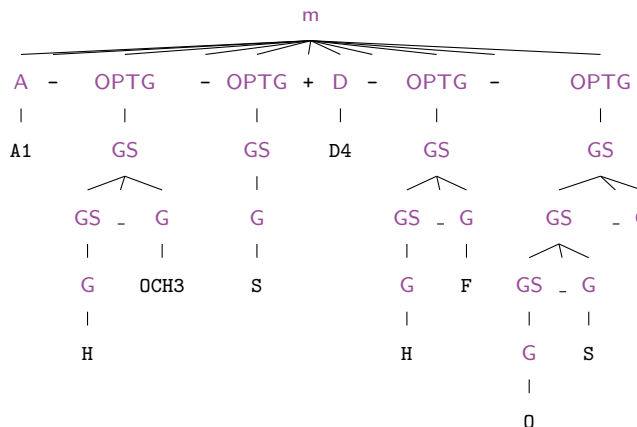


FIG. 10: Parse tree for A1-H\_OCH3-S+D4-H\_F-O\_S\_S

the original input string from the embedding  $\mathbf{z}$ .

## REFERENCES

- <sup>1</sup>M. A. Green, Y. Hishikawa, W. Warta, E. D. Dunlop, D. H. Levi, J. Hohl-Ebinger, and A. W. Ho-Baillie, Progress in Photovoltaics: Research and Applications **25**, 668 (2017), pIP-17-089.
- <sup>2</sup>M. Turbiez, P. Frère, M. Allain, C. Videlot, J. Ackermann, and J. Roncali, Chemistry – A European Journal **11**, 3742 (2005).
- <sup>3</sup>M. Cheng, X. Yang, F. Zhang, J. Zhao, and L. Sun, The Journal of Physical Chemistry C **117**, 9076 (2013), <http://dx.doi.org/10.1021/jp311378b>.
- <sup>4</sup>D. P. Hagberg, T. Marinado, K. M. Karlsson, K. Nonomura, P. Qin, G. Boschloo, T. Brinck, A. Hagfeldt, and L. Sun, The Journal of Organic Chemistry **72**, 9550 (2007), pMID: 17979286, <http://dx.doi.org/10.1021/jo701592x>.
- <sup>5</sup>T. Xu and L. Yu, Materials Today **17**, 11 (2014).

```

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

- <sup>6</sup>B.-G. Kim, X. Ma, C. Chen, Y. Ie, E. W. Coir, H. Hashemi, Y. Aso, P. F. Green, J. Kieffer, and J. Kim, *Advanced Functional Materials* **23**, 439 (2013).
- <sup>7</sup>K. B. Ornsø, J. M. Garcia-Lastra, and K. S. Thygesen, *Phys. Chem. Chem. Phys.* **15**, 19478 (2013).
- <sup>8</sup>I. Y. Kanal, S. G. Owens, J. S. Bechtel, and G. R. Hutchison, *The Journal of Physical Chemistry Letters* **4**, 1613 (2013).
- <sup>9</sup>E. O. Pyzer-Knapp, K. Li, and A. Aspuru-Guzik, *Advanced Functional Materials* **25**, 6495 (2015).
- <sup>10</sup>Y. Imamura, M. Tashiro, M. Katouda, and M. Hada, *The Journal of Physical Chemistry C* **121**, 28275 (2017).
- <sup>11</sup>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, (2017), arXiv:1702.05532 [physics.chem-ph].
- <sup>12</sup>R. Gómez-Bombarelli, D. Duvenaud, J. M. Hernández-Lobato, J. Aguilera-Iparraguirre,

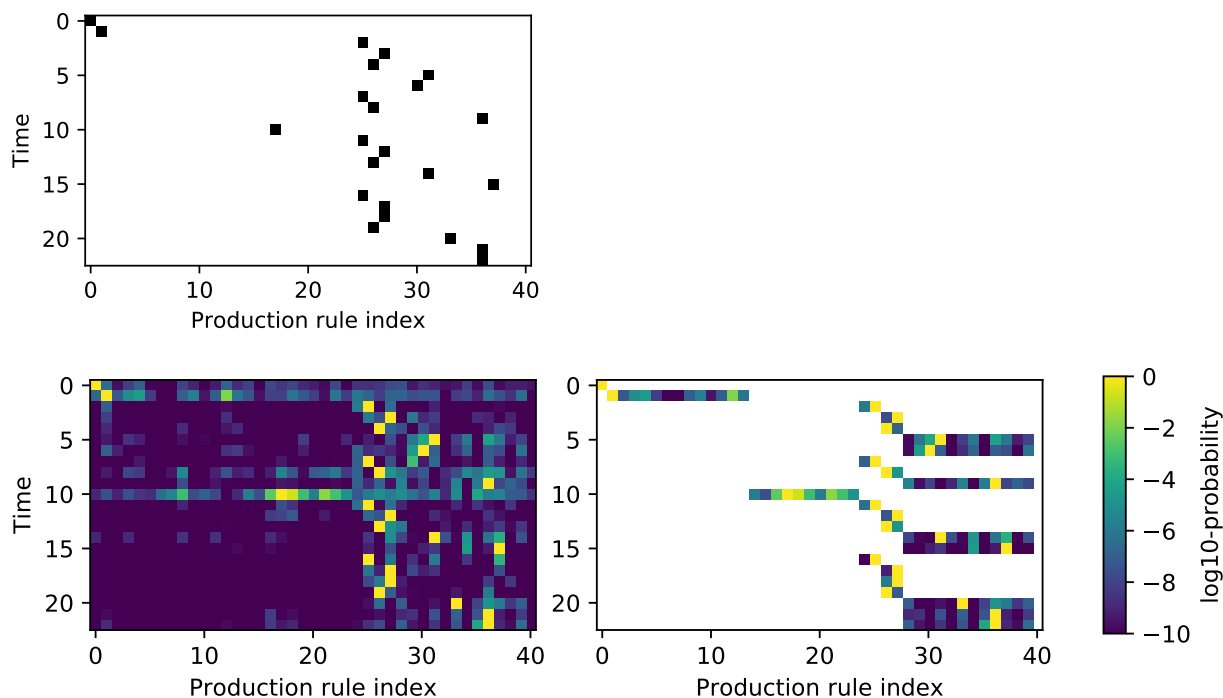


FIG. 12: Top: sequence of production rules encoded into binary matrix. Bottom left: log-probabilities assigned to production rules after each time step of decoder. Bottom right: log-probabilities assigned to production rules after applying mask.

- T. D. Hirzel, R. P. Adams, and A. Aspuru-Guzik, (2016), arXiv:1610.02415 [cs.LG].
- <sup>13</sup>M. J. Kusner, B. Paige, and J. M. Hernández-Lobato, (2017), arXiv:1703.01925 [stat.ML].
- <sup>14</sup>D. Weininger, J. Chem. Inf. Comput. Sci. **28**, 31 (1988).
- <sup>15</sup>I. Hill, A. Kahn, Z. Soos, and J. R.A. Pascal, Chemical Physics Letters **327**, 181 (2000).
- <sup>16</sup>R. S. Kularatne, H. D. Magurudeniya, P. Sista, M. C. Biewer, and M. C. Stefan, Journal of Polymer Science Part A: Polymer Chemistry **51**, 743 (2013).
- <sup>17</sup>A. Facchetti, Materials Today **16**, 123 (2013).
- <sup>18</sup>T. E. Kang, J. Choi, H.-H. Cho, S. C. Yoon, and B. J. Kim, Macromolecules **49**, 2096 (2016), <http://dx.doi.org/10.1021/acs.macromol.5b02772>.
- <sup>19</sup>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), <http://dx.doi.org/10.1021/ma501736m>.
- <sup>20</sup>E. Bundgaard and F. C. Krebs, Solar Energy Materials and Solar Cells **91**, 954 (2007), low Band Gap Polymer Materials for Organic Solar Cells.
- <sup>21</sup>L. Zhang, K. Pei, M. Yu, Y. Huang, H. Zhao, M. Zeng, Y. Wang, and J. Gao, The Journal

- of Physical Chemistry C **116**, 26154 (2012), <http://dx.doi.org/10.1021/jp306656c>.
- <sup>22</sup>N. Blouin, A. Michaud, D. Gendron, S. Wakim, E. Blair, R. Neagu-Plesu, M. Belletête, G. Durocher, Y. Tao, and M. Leclerc, *Journal of the American Chemical Society* **130**, 732 (2008), pMID: 18095689, <http://dx.doi.org/10.1021/ja0771989>.
- <sup>23</sup>L. Zhang, M. Yu, H. Zhao, Y. Wang, and J. Gao, *Chemical Physics Letters* **570**, 153 (2013).
- <sup>24</sup>R. Ramakrishnan, P. O. Dral, M. Rupp, and O. A. von Lilienfeld, *Sci Data* **1**, 140022 (2014).
- <sup>25</sup>L. Ruddigkeit, R. van Deursen, L. C. Blum, and J.-L. Reymond, *J. Chem. Inf. Model.* **52**, 2864 (2012).
- <sup>26</sup>M. Rupp, A. Tkatchenko, K.-R. Müller, and O. A. von Lilienfeld, *Phys. Rev. Lett.* **108**, 058301 (2012).
- <sup>27</sup>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).
- <sup>28</sup>B. Huang and O. A. von Lilienfeld, *J. Chem. Phys.* **145**, 161102 (2016).
- <sup>29</sup>D. Rogers and M. Hahn, *J. Chem. Inf. Model.* **50**, 742 (2010).
- <sup>30</sup>K. T. Schütt, P.-J. Kindermans, H. E. Saucedo, S. Chmiela, A. Tkatchenko, and K.-R. Müller, (2017), [arXiv:1706.08566 \[stat.ML\]](https://arxiv.org/abs/1706.08566).
- <sup>31</sup>J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, (2017), [arXiv:1704.01212 \[cs.LG\]](https://arxiv.org/abs/1704.01212).
- <sup>32</sup>D. P. Kingma and M. Welling, (2013), [arXiv:1312.6114v10 \[stat.ML\]](https://arxiv.org/abs/1312.6114v10).
- <sup>33</sup>D. J. Rezende, S. Mohamed, and D. Wierstra, in *Proceedings of The 31st International Conference on Machine Learning* (2014) pp. 1278–1286.
- <sup>34</sup>J. J. Irwin, T. Sterling, M. M. Mysinger, E. S. Bolstad, and R. G. Coleman, *J. Chem. Inf. Model.* **52**, 1757 (2012).
- <sup>35</sup>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).
- <sup>36</sup>D. Janz, J. van der Westhuizen, B. Paige, M. J. Kusner, and J. M. Hernandez-Labato, (2017), [arXiv:1712.01664 \[stat.ML\]](https://arxiv.org/abs/1712.01664).
- <sup>37</sup>H. Dai, Y. Tian, B. Dai, S. Skiena, and L. Song, in *NIPS 2017 Workshop Machine Learning*

- for Molecules and Materials* (2017).
- <sup>38</sup>M. Simonovsky and N. Komodakis, (2018), arXiv:1802.03480 [cs.LG].
- <sup>39</sup>J. You, R. Ying, X. Ren, W. L. Hamilton, and J. Leskovec, (2018), arXiv:1802.08773 [cs.LG].
- <sup>40</sup>Y. Li, O. Vinyals, C. Dyer, R. Pascanu, and P. Battaglia, (2018), arXiv:1803.03324 [cs.LG].
- <sup>41</sup>K. Wu, N. Sukumar, N. A. Lanzillo, C. Wang, R. “Rampi” Ramprasad, R. Ma, A. F. Baldwin, G. Sotzing, and C. Breneman, *Journal of Polymer Science Part B: Polymer Physics* **54**, 2082 (2016).
- <sup>42</sup>F. Pereira, K. Xiao, D. A. R. S. Latino, C. Wu, Q. Zhang, and J. Aires-de Sousa, *Journal of Chemical Information and Modeling* **57**, 11 (2017), pMID: 28033004, <http://dx.doi.org/10.1021/acs.jcim.6b00340>.
- <sup>43</sup>“Pubchem substructure fingerprint,” [ftp://ftp.ncbi.nlm.nih.gov/pubchem/specifications/pubchem\\_fingerprints.txt](ftp://ftp.ncbi.nlm.nih.gov/pubchem/specifications/pubchem_fingerprints.txt) (2017), accessed: 2017-11-21.
- <sup>44</sup>T. A. Halgren, *Journal of Computational Chemistry* **17**, 616 (1996).
- <sup>45</sup>N. M. O’Boyle, M. Banck, C. A. James, C. Morley, T. Vandermeersch, and G. R. Hutchison, *Journal of Cheminformatics* **3**, 33 (2011).
- <sup>46</sup>W. Kohn and L. J. Sham, *Phys. Rev.* **140**, A1133 (1965).
- <sup>47</sup>A. D. Becke, *The Journal of Chemical Physics* **98**, 5648 (1993).
- <sup>48</sup>A. Schäfer, C. Huber, and R. Ahlrichs, *The Journal of Chemical Physics* **100**, 5829 (1994).
- <sup>49</sup>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,” (2009).

- <sup>50</sup>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,” (2018), unpublished journal article.
- <sup>51</sup>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), <http://dx.doi.org/10.1021/ma4005023>.
- <sup>52</sup>J. Torras, J. Casanovas, and C. Alemán, *The Journal of Physical Chemistry A* **116**, 7571 (2012).
- <sup>53</sup>R. E. Larsen, *The Journal of Physical Chemistry C* **120**, 9650 (2016).
- <sup>54</sup>K. T. Schütt, F. Arbabzadah, S. Chmiela, K. R. Müller, and A. Tkatchenko, *Nat. Commun.* **8**, 13890 (2017).
- <sup>55</sup>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, *Journal of Machine Learning Research* **12**, 2825 (2011).
- <sup>56</sup>J. Besag, *Journal of the Royal Statistical Society. Series B: Methodological* **B-48**, 259 (1986).
- <sup>57</sup>Theano Development Team, (2016), [arXiv:1605.02688](https://arxiv.org/abs/1605.02688) [cs.SC].
- <sup>58</sup>D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling, (2016), [arXiv:1606.04934](https://arxiv.org/abs/1606.04934) [cs.LG].
- <sup>59</sup>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).
- <sup>60</sup>S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Jozefowicz, and S. Bengio, (2015), [arXiv:1511.06349](https://arxiv.org/abs/1511.06349) [cs.LG].
- <sup>61</sup>K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, (2014), [arXiv:1406.1078](https://arxiv.org/abs/1406.1078) [cs.CL].
- <sup>62</sup>N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, *J. Mach. Learn. Res.* **15**, 1929 (2014).