Deciphering the Contents of Chemically-Trained Neural Networks into Physical Intuition

Eric Berquist

June 15th, 2017

1 Overview and Objectives

The explosive growth of computing power has enabled the use of machine learning (ML) models for the accurate calculation of chemical properties. Current ML models are capable predicting energetic properties of small, neutral organic molecules to chemical accuracy (1 kcal/mol).[1] However, it is unclear *what* the quality of a prediction will be when a ML model is given a molecule outside its training set, and *why* the model is giving that prediction. Each ML model is a set of unknown parameters that constitute a black box,[2] where it is not known or understood *how* the model functions, *why* the model gives the predictions it does, and whether or not correct predictions are being made for the right reasons.[3] A critical step for the continued application of ML models to quantum chemistry will be to make predictions beyond the relatively simple example systems that are currently seeing widspread use.[4] Some of the most important, interesting, and unexplored chemistry in terms of reactivity and intrinsic molecular properties involves species that are either charged or have unpaired electrons, both of which are outside the scope of current applications. If a model is not right for the right reasons, there cannot be an expectation that it is transferable to even similar inputs, and is an indicator that the architecture is insufficient for training on more complex datasets. Such a battle is being fought within electronic structure theory,[5–8] with the most negative potential consequence being invalidation of the past decade's application of quantum chemical methods to chemical problems.

The overall objective is to quantify what quantum chemistry-trained machine learning models are actually learning. The central hypothesis is that the application of machine learning to quantum chemistry is physically motivated, *i.e.*, ML models are learning parameters that connect the molecular representation (input features) to predictions in a manner that is qualitatively identical to how a trained chemist would apply their intuition. An implicit connection has already been made[1] between how a molecule is represented as input to ML models and the accuracy of prediction for certain kinds of properties, but this connection has not been explored. The rationale is that by understanding the connection between the molecular representation given to ML models and their predictions, ML models themselves can be used for rationalizing molecular structure-property relationships. The overall objective will be achieved through the following specific aims:

1. **Reproduce existing machine learning models for molecular properties from the literature.** I will replicate two neural networks (NNs) from [1] to serve as both the necessary architecture that analysis tools developed later in this proposal will be built on, and as the baseline for prediction accuracy. NNs are chosen since they overall exhibit the most quantitatively accurate predictions compared to other ML architectures when trained on quantum chemical data. I will reproduce the isotropic static polarizability $\bar{\alpha}$ [9] and the zero-point vibrational energy E_{ZPVE} [10], which represent molecular properties that can be derived from spectroscopic observables and are difficult to predict with current NN approaches.

2. **Characterize the parameters learned by existing machine learning models from the literature.** I will use relevance propagation[11–13] to quantify the relative importance of each input feature for each predicted property of the NNs in aim #1. Relevance propagation gives insight into a model's learned coefficients in a fashion that can be used directly in statistical analysis and visualization.

3. **Train supervised neural networks on complex molecular properties.** Using the molecular database from aim #1, I will predict their vibrational and nonlinear optical properties, specifically the static parallel first hyperpolarizability $\beta_{\parallel}$ [14] and the set of frequencies $\{\tilde{\nu}\}$ that compose stick vibrational spectra, as they represent a natural extension in complexity from $\bar{\alpha}$ and E_{ZPVE}.

4. **Characterize the parameters learned for complex molecular properties using unsupervised neural networks.** I will use the relevance propagation techniques developed in aim #2 to show the underlying causes for the performance of the NNs trained in aim #3. To aid in quantifying the importance of molecular features to property predictions, I will train a denoising autoencoder-based neural network in an unsupervised manner to provide a compressed representation of molecular inputs. Unsupervised NNs are capable of discovering intrinsic properties of their input samples,[15] which will provide a summarized interpretation of molecular structure importance when combined with relevance propagation.

The expected outcomes are

- clear connections between the input molecular representations and predicted outputs that can be used to build quantitative structure-function relationships, that
- adding an unsupervised learning stage provides a reduced-dimensionality molecular representation connecting directly to chemical intuition,
- well-defined and publicly-available protocols for applying more complex ML models to chemical properties, regarding all stages of the prediction pipeline: generation of the QM data from calculations, preparation of the input, training of the model, and analysis of the results, and
- ML models capable of predicting complex molecular properties, with the first proof-of-concept predictions of higher-order nonlinear optical properties and vibrational spectra.

2 Significance

This is the first attempt at understanding the parameters of ML models used to predict microscopic and macroscopic molecular properties, rather than treating the models as black boxes that cannot be understood. If ML models are shown to be learning physically-motivated or chemically-intuitive parameters, ML can become more than just a path for the accurate prediction of molecular properties, but be transformed into a tool itself that can give deep insight into molecular structure-property relationships. ML models are expensive to train, maintain, and extend in terms of both human and CPU time.[16] If this proposal shows that ML models are not learning properly, then the scientific research community can avoid the wasteful use of resources on model training and shift their focus to developing better model architectures than the current state-of-the-art.

3 Review of Relevant Literature

3.1 Machine Learning

Machine learning is the ability for computers to "learn" without being given explicit instructions. Rather than providing exact instructions though traditional programming, computers are fed sets of input data and are usually expected to return a certain result. By training itself to reproduce results, a learned ML model would ideally be able to predict outputs for new, unknown inputs. Common applications of ML are in email spam filtering, search engine prediction, image and voice recognition, and self-driving cars.

Some definitions and terms used throughout this proposal are

- *Architecture*: the formal structure of the network or ML model itself, encompassing the region from equations and diagrams to the implementation (code).
- *Model*: an architecture implemented in code with learned parameters.
- *Pipeline*: multiple steps and components chained together, such as the preparation of data for input into an architecture, the architecture itself, and any steps required to transform the architecture output into something else useful, such as visualizations or statistics.

The two types of learning used in this proposal are

- *Supervised learning*: Train a machine learning model using data where the correct output prediction is known and given for each input sample, and the goal of the model is to predict similar types of outputs for new inputs.
- *Unsupervised learning*: Train a machine learning model using data where the correct output prediction is not given, and the goal of the model is to learn intrinsic properties of the inputs by recreating the input as output.

3.2 Machine Learning in Chemistry

The use of machine learning to make chemical predictions is not new, with work dating back over 25 years for prediction of NMR spectra using small neural networks trained on experimental data.[17] The largest application of machine learning to chemical problems is within cheminformatics, where it has seen wide use within industrial drug discovery with emphasis on predicting quantitative structure-activity relationships (QSAR).[18] The goal is to predict the activity of a given drug candidate based on experimental activities of many other molecules, with inputs being information about atom types, bond types, number of aromatic rings, atomic partial charges, and other pieces of structural information, all of which are related to the molecular graph or connectivity.[19]

In particular, there is a recent application of deep neural networks (DNNs) to QSAR datasets,[20] which contains a systematic study for determining the best model parameters. The machine learning community calls this "hyperparameter tuning", which is another term for parameter optimization. However, this is still an empirical black-box approach, where the input is manually manipulated and statistical analysis is performed on the output, but this does not provide enough insight into how or why the quality of a model changes. For example, whether or not a rectified linear unit (ReLU) or sigmoid unit is the best function to represent neuron activation says nothing about why one molecule may be more potent than another in a QSAR study. This brute-force type of parameter optimization *does* provide a good starting point for understanding the sensitivity of a ML model. Unfortunately, even parameter optimization has not been extensively performed on models trained using quantum chemical data. In that sense, cheminformatics is a step ahead of other sub-disciplines in chemistry regarding the *application* of machine learning models, but not in the *understanding* of their models.

Their parameter optimization study is especially relevant to this proposal because it examines the effect of placing an unsupervised NN before other NNs for unsupervised pre-training. Surprisingly, the authors found that an unsupervised pretraining step decreased the accuracy of their predictions, which is counter to the expected outcome of this proposal. However, the paper implies that their results are not even valid due to algorithmic restrictions in their software. Therefore it seems incorrect to draw any conclusions from this, such as "no unsupervised pretraining is needed". It would be interesting to see if the same conclusion is drawn for models trained on quantum chemical data using the proper algorithms.

Additionally, it is unclear why a DNN trained on the combination of all QSAR datasets (called a "joint DNN") performs better than separate DNNs for each dataset when considering the lack of overlap in the training sets. The methods developed in this proposal, while being applied to models trained on quantum chemical data, should be applicable to any DNN (consider that relevance propagation is mostly developed in computer vision/image recognition). One goal of this proposal is to transfer the idea of relevance propagation from its original intended application field to other fields. If it is indeed transferable, then it may shed some light on why unsupervised learning resulted in decreased prediction performance and the improvement of join DNNs over separate DNNs.

3.3 Machine Learning in Quantum Chemistry

However, the use of machine learning in quantum chemistry, specifically electronic structure theory, is relatively new, with the earliest references on Scopus dating back to 2008, a large spike in 2013, and rapid growth from 2015 to today. The goal is to predict more elemental properties than within cheminformatics, such as the internal energy, enthalpy, free energy, heat capacity, HOMO and LUMO energies and gaps, dipole moments, polarizabilities, and zero-point vibrational energies.[21] More advanced applications are the use of neural networks for predicting the products of organic reactions[22] and the transport properties of candidates for organic photovoltaics.[23]

There have not been attempts to predict tensorial properties, just scalar-valued properties. This precludes the prediction of full spectroscopic properties, which are mathematically not representable as single scalars. There is recent work considering the prediction of full spectra, specifically linear vibrational spectra from *ab initio* molecular dynamics (AIMD) simulations.[24] However, this proposal is concerned with the generation of spectra from static calculations, which avoids some convolution of the calculated spectra being dependent on the model's learned representation of the potential energy surface. Additionally, their vibrational spectra were calculated from the dipole autocorrelation function, which is dependent on artificially partitioning the electron density into atomic charges, which they derive from the neural network. Thus, this is not an end-to-end[25] prediction of molecular spectra from a single structure, as will be performed in this proposal. By performing end-to-end prediction rather than decomposing the problem so that the neural network only considers part of the prediction task, this proposal pushes the limits of attempting to train neural networks on the prediction of complex molecular properties.

3.4 Relevance Propagation

Layer-wise relevance propagation (LRP, or relevance propagation) is a method for identifying what a ML model has learned[11] in terms of the model's input features. Figure 1 is a concrete example of what the output from relevance propagation looks like when applied to image classification by a neural network. Here, we assume that the network correctly identified the subject of the image as a cat (rather than a dog or a potted plant), but relevance propagation shows which image pixels were most important for the network to determine the photo is of a cat. The pixel-wise importance is a single number for each pixel that can be interpreted as a contribution for that pixel to the final classification of the image. More generally, is it the relative importance of each input feature to the predicted output; here and in other image recognition examples, pixels are input features. Applications to image classification resulting in pixel importance naturally lends itself to visualizing the output as a heatmap on top of the original input.

Other methods exist for assigning rules of how input features map to predictions.[3,26,27] Several of these are based on the idea of gradient perturbations, where repeated changes in prediction are measured as a result of small changes in the input. Performed enough times, this creates a map of the network's decision boundary.[28] A gradient perturbation-based method is unsatisfactory because it requires repeated forward passes through the network with a set value for the perturbation size, and relevance propagation requires only

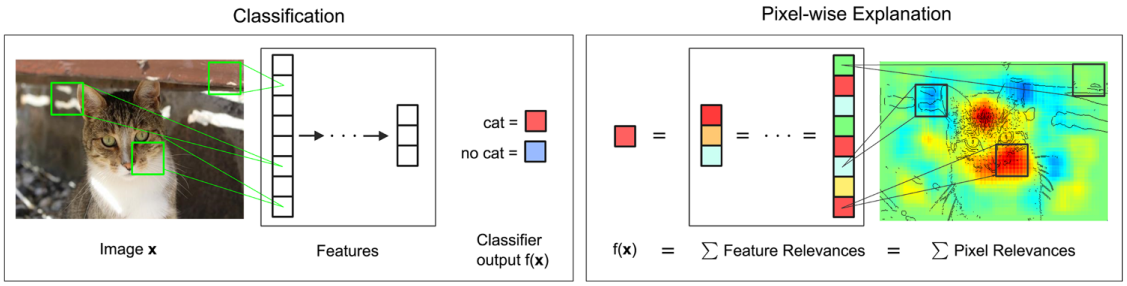

Figure 1: Example of output from relevance propagation showing which sections of an image the neural network considered important during classification. Taken from [11].

one backward pass with a single free but insensitive parameter. Additionally, most methods for assigning input relevance have only been used for image classification, where the input features are of uniform type (pixel data). The input featurization for representing molecular structures[1] is heterogeneous, and it is unclear how the perturbation parameter should be varied for each kind of molecular feature.

The closest use of input relevance in molecular predictions is monitoring the evolution of input features as network training progresses,[19] shown in 2. However, there is no definition for what the evolution of input features is, such as the metric for evolution, or what the units are. One aim of this proposal is to form a quantitative basis connecting molecular features to model predictions that figures such as 2 can be built upon.

Although no improvements will be made to the basic relevance propagation algorithm itself, there is novelty in two areas. To the best of the PI's knowledge, this is the first time relevance propagation will be applied to a regression task rather than a classification task, and the first time relevance propagation will be applied outside of image classification.

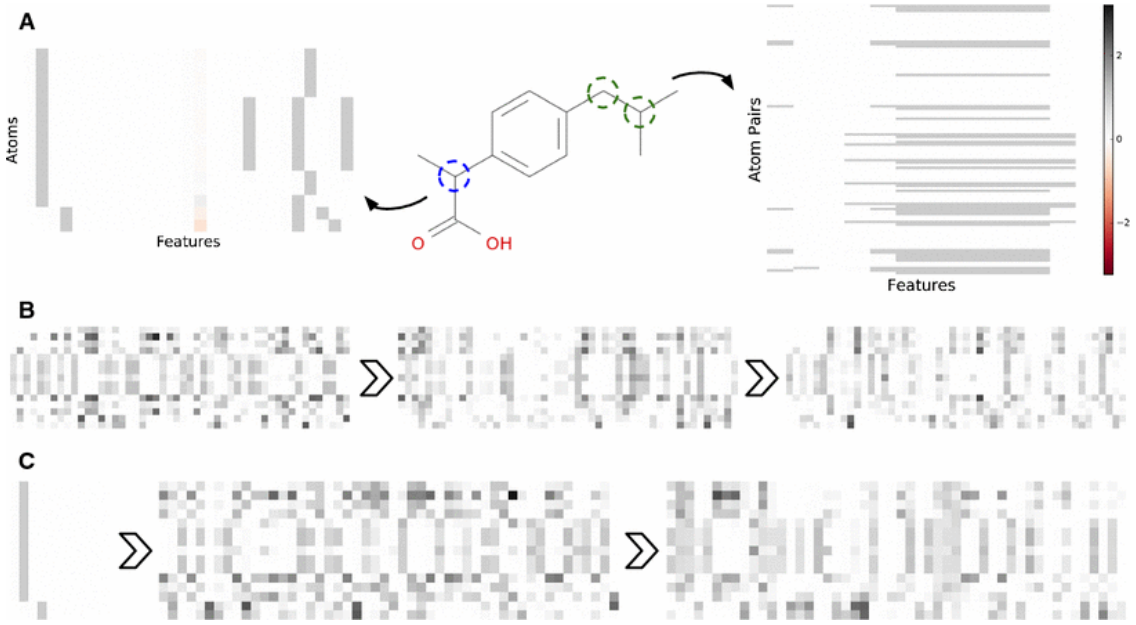

Figure 2: Example of how a molecule is mapped to input features, and how the input features change as training epochs pass. Taken from [19].

3.5 Unsupervised Training of Neural Networks

Prior applications of NNs to quantum chemistry primarily used supervised training, with the goal of predicting a molecular energy or property with increasing accuracy. The supervised training of NNs requires a separate quantum chemical calculation on each input for each target property, which is not tenable for databases sizes that represent meaningful chemical space (such as PubChem[29]) or any high-level calculation methods (such as coupled cluster).

An original use of unsupervised learning is for "pretraining", where the bottom layers in a NN are trained to reproduce the inputs before switching to supervised learning. Currently, unsupervised pretraining is not commonly used within the image classification community due to the discovery of better weight initialization and structural choices for the network (ReLUs for the activations, convolutional rather than directly-connected layers), rendering it unnecessary. In the future, as the both the size of available training sets (here, chemical databases) and NN architectures grows, unsupervised pretraining may take a more prominent role.

Unsupervised training of NNs is not without precedent in quantum chemistry; generative adversarial networks (GANs) have been used to generate "hallucinated" (fake or imperfect) representations of methanol trimers, which are then fed into a fully-connected NN to predict the three-body energy.[30] Although the error is ~2× that of the standard Coulomb matrix-based NN, this may be due to the use of more convolutional than fully-connected layers in the GAN.

Rather than use unsupervised learning for network pretraining, I will use the dimensionality reduction capability of a denoising autoencoder[15] with only fully-connected layers to generate features that will be identified using layer-wise relevance propagation. The rendered hallucinations from the GANs (depth maps or D-maps) also provide an interesting connection back to LRP for future work.

4 Research Plan

4.1 Specific Aim #1: Reproduction of Existing Literature Neural Networks

4.1.1 Introduction

The objective is to reproduce trained neural networks from the literature in order to create the foundation of the machine learning pipeline to be developed within this proposal and to serve as a validation baseline for further predictions. The hypothesis is that published quantum chemical neural networks are entirely reproducible by connecting free, open-source tools. To test this hypothesis, I will reproduce the ML pipeline and results from [1], specifically the isotropic static polarizability $\bar{\alpha}$ and the zero-point vibrational energy E_{ZPVE}. The expected outcome is a fully-worked and documented reproduction of neural networks from the literature that can serve as the basis for future pipelines within the wider chemistry and machine learning communities.

4.1.2 Research Design

Unfortunately, the learned models for the results presented in [1] are not available, only descriptions of the architectures. Recreating the literature models requires an implementation of the model architecture and input data in the proper format.

There are two neural network-based architectures described in [1]: Graph Convolutions[19] (GC) and Gated Graph Neural Networks[31] (GG). These NN architectures are used again as baselines in [32]. Since the original GC implementation referenced in [1] is openly available,[33] I will use the GC-based architecture with modifications described in section E5 of [1]. Details for the GC architecture input, called the Molecular Graph representation, are shown in tables 1 and 2 of [1] and reproduced here.

Table 1: The Molecular Graph (MG) input representation: single atom features

Feature	Description	Size
Atom type	H, C, N, O, or F (one-hot)	5
Chirality	R or S (one-hot or null)	2
Formal charge	Integer electronic charge	1
Partial charge	Calculated partial charge	1
Ring sizes	For each ring size (3-8), the number of rings that include this atom	6
Hybridization	sp, sp^2, or sp^3 (one-hot or null)	3
Hydrogen bonding	Whether this atom is a hydrogen bond donor and/or acceptor (binary values)	2
Aromaticity	Whether this atom is part of an aromatic system	1
		21

Table 2: The Molecular Graph (MG) input representation: atom pair features

Feature	Description	Size
Bond type	Single, double, triple, or aromatic (one-hot or null)	4
Graph distance	For each distance (1-7), whether the shortest path between the atoms in the pair is less than or equal to that number of bonds (binary values)	7
Same ring	Whether the atoms in the pair are in the same ring	1
Spatial distance	The Euclidean distance between the two atoms	1
		13

The QM9 dataset consists of 134K molecules[4] containing up to 9 heavy atoms from the elements C, N, O, and F, with a maximum number of 29 atoms. The representation in tables 1 and 2 will result in an input size of $21\binom{x}{1}+13\binom{x}{2}$ for a given number of atoms x, leading to a total length of 5,887 for the maximum number of 29 atoms in QM9. Inputs are available as modified XYZ files from the Quantum Machine website under the QM9 Dataset section,[4,34] which will be transformed into the Molecular Graph (MG) representation using RDKit[35] with Gasteiger partial charges as in [19]. I will then modify the Graph Convolutions architecture from [19] as described in Section E5 of [1]. Using the model parameters described in that

section, I will train two separate models, one for the isotropic static polarizability $\bar{\alpha}$, and another for the zero-point vibrational energy E_{ZPVE}.

For the reproduction of literature results, the only numerical values from [1] are in Table 3, which shows the mean absolute error (MAE) for each input representation-architecture combination. Because the sample size of QM9 is sufficiently large (134K molecules), the MAE is calculated using out-of-sample validation, where the ML models are trained using 90% of the available data and compared against the DFT (B3LYP/6-31(2df,p)) results for the remaining 10%. The 90% constitutes ~117K molecules after removing 3K from 134K due to failed SMILES consistency tests. This 90/10 (training + validation)/test set split allows for 10-fold cross-validation. It is not mentioned how the concrete splits are obtained or how the final MAE is calculated from the 10 models. For this proposal, I will perform an unbiased random shuffle of QM9 index codes and split them into 10 uniform bins. After training and model validation using the procedure described above, the final MAE will be calculated as the mean of the 10 individual MAEs. The literature models will be considered replicated if the two final models have MAEs within 95% of 0.227 a_0^3 for the polarizability and 0.00975 eV for the ZPVE, respectively.

4.1.3 Expected Outcomes

The concrete products of this aim will be a set of Python scripts that transform the XYZ-like files into the Molecular Graph representation, implement the modified Graph Convolutions architecture, train MG/GC models for each molecular property ($\bar{\alpha}$ and E_{ZPVE}) using out-of-sample cross-validation, and calculate each molecular property from the trained models when given a normal XYZ molecular structure. This will be a fully-worked and documented reproduction of neural networks from the literature that can serve as the basis for not only this proposal's later aims, but for future pipelines within the wider chemistry and machine learning communities. These scripts will take the form of Jupyter Notebooks,[36,37] which combine code, math, and documentation in an easy-to-replicate package that is popular in the machine learning community.

4.2 Specific Aim #2: Characterization of Existing Literature Neural Networks

4.2.1 Introduction

The objective is to quantify what already-published neural network-based ML models have learned. The hypothesis is that when predicting an output, the most important (relevant) parts of the input for that output align with our trained chemical intuition. Specifically, for strongly geometry-dependent properties, such as the ZPVE, more relevance will be placed on geometric input features such as bond lengths and bond types. For strongly wavefunction- or density-dependent properties, such as the isotropic polarizability, more relevance will be placed on electronic input features such as partial atomic charges compared to other features. To test this hypothesis, I will develop the necessary ML pipeline for adding relevance propagation and analysis steps to the already-published ML models. This will involve connecting existing relevance propagation tools[13,38–40] to the end of the pipeline from aim #1

and creating a human-understandable representation of the relevance propagation output in terms of molecular features.

4.2.2 Research Design

The authors of the relevance propagation algorithm I will use have created an open-source reference implementation in Python. From their GitHub page:[38]

> The Layer-wise Relevance Propagation (LRP) algorithm explains a classifier's prediction specific to a given data point by attributing relevance scores to important components of the input by using the topology of the learned model itself. The LRP Toolbox provides simple and accessible stand-along implementations of LRP for artificial neural networks supporting Matlab and Python.

This reference implementation is interfaced with its own implementation of composable neural networks. This is not immediately compatible with the TensorFlow-based implementation of the Graph Convolutional neural networks trained in aim #1. However, there is open-source initial interface code available for connecting TensorFlow-based models with LRP.[39,40] I will use these three implementations to write a Python interface between the trained GC models and LRP.

Once the LRP implementation is connected to the trained models, details of running the LRP algorithm must be considered. There are multiple decomposition variants of LRP, each with different trade-offs regarding numerical stability and conserving relevance.[12] Because the number tunable parameters is small, with only one for each of the three variants, and there are examples of modulating these parameters in the literature, I will start with the variant that requires no free parameters and extend to the other variants if results do not make sense.

All examples from the relevance propagation literature focus on explaining image classification decisions using heat maps overlaid on the input image decision boundary. Instead of performing image classification, the neural networks trained in aim #1 perform regression using a graph-type, heterogeneous input, so a different form of interpreting the results will be needed. LRP produces output on a per-sample basis with a signed relevance value for each input feature, where the sum of all relevance values equals the prediction output. For each molecule in the QM9 dataset, I will run the LRP implementation and perform the equivalent of feature normalization and scaling so that the relevances have zero mean with a minimum and maximum corresponding to the largest absolute values in the QM9 dataset. Not all molecules in QM9 have the same number of atoms, so for each molecule I will average together the relevances over all atoms to produce a single relevance for each type of feature (12 total). While this can be viewed as a significant loss of information, the goal of this proposal is to identify what types of features are generally important for different molecular properties, not the quality of a prediction for a specific molecule.

4.2.3 Expected Outcomes

Once LRP is applied to the trained models, TODO

Additionally, the LRP analysis will be attached to the pipeline developed in aim #1, with complete documentation and code that replicates the work from this proposal.

4.3 Specific Aim #3: Training Neural Networks for Complex Molecular Properties

4.3.1 Introduction

The objective is to see how currently used neural network architectures perform for more complex molecular properties than those found in the literature. The hypothesis is that the more complex properties $\beta_{\|}$ and $\{\tilde{\nu}\}$ are expected to have larger relative errors compared to $\bar{\alpha}$ and E_{ZPVE}, in particular the set of vibrational frequencies, as predictions of the highest fundamental frequency ω_1 alone already have large errors.[1] Testing this hypothesis first requires calculating the hyperpolarizability for each of the training samples, followed by training the network itself.

4.3.2 Research Design

Results for $\{\tilde{\nu}\}$ are already present as labeled data in the QM9 dataset.[4] I will use the Dalton quantum chemistry program package[41] for the hyperpolarizability calculations, as it is free for academic use and designed especially for the calculation of molecular response properties such as hyperpolarizabilities. These calculations will employ the B3LYP density functional in combination with the 6-31G(2df,p) basis set to maintain comparability with past calculations from the QM9 dataset.[4]

Training the NN-based model for the hyperpolarizability will not require any modifications to the NN architecture, as $\beta_{\|}$ is a scalar quantity. As the set of vibrational modes is dependent on molecular size, the architecture that will be trained to reproduce $\{\tilde{\nu}\}$ will be modified to output the maximum possible number of vibrational frequencies, with the first $3N-6$ output nodes returning predictions and the remainder padded with zeros. This is a well-known limitation of many ML architectures using variable-dimension input and output.

The error metric used for the set of vibrational frequencies will be identical to that for ω_1, which has an MAE of 3.15 cm^{-1}. The error in harmonic vibrational frequencies is not uniformly distributed among all normal modes [REF], but comparing the literature error for ω_1 against the same value trained on a network for all vibrational frequencies is a measure of how a joint NN compares to a separate NN for each vibrational frequency. For the hyperpolarizability, the MAD of the DFT calculations used in training will be used as the baseline for the MAE of the trained NN prediction. In both cases, since training is occurring on novel properties rather than replicating literature values, achieving DFT accuracy is the goal for training convergence.

4.3.3 Expected Outcomes

Similar to aim #1, the result will be a full set of documented Jupyter Notebooks that parse the hyperpolarizability calculation outputs and train separate MG/GC models for the hyperpolarizability and vibrational frequencies. In addition to the trained models, I will provide the raw calculation outputs for the hyperpolarizability calculations on figshare,[42] which is capable of hosting large archives. These outputs will contain full dipole, polarizability, and hyperpolarizability tensors, along with other common quantities from quantum chemical calculations such as Löwdin partial atomic charges and MO coefficients. Distributing these

unmodified outputs enables the future prediction of even more complex properties and the use of molecular properties in feature generation.

4.4 Specific Aim #4: Characterization of Novel Neural Networks

4.4.1 Introduction

The objective is to determine the relative importance of each component in the molecular representation to predictions of complex molecular properties. The hypothesis is that the most important input features for $\beta_{\parallel}$ and $\{\tilde{\nu}\}$ are similar to those for $\bar{\alpha}$ and E_{ZPVE}, respectively. This will be done by applying the analysis techniques developed in aim #2 to the neural networks trained in aim #3, along with using unsupervised learning to discover a simpler molecular representation.

4.4.2 Research Design

The type of network I will train is a denoising autoencoder (DAE), where the input is constricted down to a compressed representation as a form of dimensionality reduction (encoding) and then expanded back again to the full input size (decoding), with small amounts of random Gaussian noise is added to the input during training. For simplicity of training, it will consist of five total layers: the two input layers on either end, a coding layer in the center, and a hidden layer between the coding layer and each input layer. This will be a symmetric structure where the weights of the encoding and decoding sections are tied, meaning they are constrained to be identical, making training easier. Since the DAE will be connected to the NNs from aims #1 and #3, the input layer dimension must be same as for the GC-based models. I will choose the dimensionality of the coding layer to be twice the total number of feature types and the number of nodes in each hidden layer to be twice that, giving a 5889-136-68-136-5889 shape, with ReLU activations for each node and direct connections between each layer. This extreme constriction xxx

I will adapt the Molecular Graph input from aim 1 on top of this denoising autoencoder as implemented in TensorFlow.[43–45] Because a DAE reconstructs its inputs and does not make a classification or regression prediction, only one model needs to be trained, not four. Once the DAE model is trained, it will replace the input layer for each of the trained supervised networks. If the DAE is capable of performing good input reconstruction, then the prediction performance of these combined networks should not be much worse than the supervised networks for the training, but may be improved on validation and testing. In general, performance may decrease because the reconstruction of the original input cannot be perfect, which would constitute overfitting.

Could LRP be performed on the DAE itself? Could be another test of quality for the input reconstruction

The purpose of adding the autoencoder structure is that using layer-wise relevance propagation will show the relevance of the *encoded* input. The expectation is that because the learned encoding retains only important features for the input, xxx rather than using the possibly overcomplete input feature description. If the DAE is good at reconstructing the inputs

The original purposes of generative pretraining are to prevent overfitting in large neural networks and learn structure in large amounts of unlabeled data.

Larger databases already exist[29]

In the future, with larger amount of unlabeled data (we don't want to have to run millions of hyperpolarizability calculations in order to train), such as in [29],

In the future, it may make sense to unfreeze the autoencoder weights and change them slightly via backpropagation, so the autoencoder learns a redudced-dimensionality encoding that is more optimal for the network output and not just reconstructing the input.

4.4.3 Expected Outcomes

For this specific aim,

As a result of this proposal, I expect that the parameters learned by ML models, and therefore their predictions, will show a strong dependence on the input features in chemically-intuitive manner. This would mean that the models trained here and similar models are "right for the right reasons", and that neural network-based ML architectures are a valid path forward for predicting novel and more complex chemical properties.

5 Broader Impacts

A crucial reason for the growth in cross-disciplinary applications of machine learning is the openness and extensiveness of introductory tools, specifically tutorials and examples. Historically, chemistry lags behind other sciences in terms of openness of procedures and results. The current infrastructure surrounding the combination of machine learning and quantum chemistry is very poor: disorganized work, disorganized results, and not all components are available for reuse. The development of these machine learning pipelines will constitute the development of open-source, freely available infrastructure that will be easily extendable. I will provide openly **all** components of the machine learning pipeline developed in this proposed work, including the fully-trained models, meaning the implementations using open-source software and the learned parameters for each model. All components will be placed on GitHub, the premier location for the open hosting of machine learning tools. Making these tools available will enable the verification of future, more advanced machine learning models that has not been possible to date. The tools will also serve as a pedagogical example for how machine learning can be applied to quantum chemical problems.

As the application of machine learning within quantum chemistry is relatively new and fast-moving, still being in the "discovery" phase, there have not been attempts at replicating machine learning pipelines, peer-reviewed or otherwise. Additionally, in traditional quantum chemistry there are a plethora of well-known program packages for performing electronic structure calculations[41,46,47] that are self-contained, *e.g.* a single program can calculate optimized geometries, vibrational spectra, NMR chemical shifts, reaction energies, etc. This infrastructure exists to some degree for machine learning, with base packages such as scikit-learn[48] and TensorFlow[43] themselves being self-contained with excellent tutorials and examples, however this infrastructure does not exist for quantum chemistry-derived machine learning models. Introductions to machine learning are numerous and extensive using the

standard "fruit fly" of NNs, the MNIST database of handwritten digits,[49] and similar fully-worked introductions should exist for quantum chemistry as well. Releasing the pipeline from this proposal allows it to serve as the "fruit fly" for quantum chemistry in machine learning.

References

(1) Faber, F. A.; Hutchison, L.; Huang, B.; Gilmer, J.; Schoenholz, S. S.; Dahl, G. E.; Vinyals, O.; Kearnes, S.; Riley, P. F.; Anatole von Lilienfeld, O. *ArXiv e-prints* **2017**.

(2) https://en.wikipedia.org/wiki/Black_box (accessed 03/08/2017).

(3) Slavin Ross, A.; Hughes, M. C.; Doshi-Velez, F. *ArXiv e-prints* **2017**.

(4) Ramakrishnan, R.; Dral, P. O.; Rupp, M.; von Lilienfeld, O. A. *Scientific Data* **2014**, *1*, DOI: 10.1038/sdata.2014.22.

(5) Zhao, Y.; Truhlar, D. G. *Theoretical Chemistry Accounts* **2008**, *120*, 215–241, DOI: 10.1007/s00214-007-0310-x.

(6) Medvedev, M. G.; Bushmarinov, I. S.; Sun, J.; Perdew, J. P.; Lyssenko, K. A. *Science* **2017**, *355*, 49–52, DOI: 10.1126/science.aah5975.

(7) Kepp, K. P. *Science* **2017**, *356*, 496–496, DOI: 10.1126/science.aam9364.

(8) Medvedev, M. G.; Bushmarinov, I. S.; Sun, J.; Perdew, J. P.; Lyssenko, K. A. *Science* **2017**, *356*, 496–496, DOI: 10.1126/science.aam9550.

(9) Soscún, H.; Alvarado, Y.; Hernández, J.; Hernández, P.; Atencio, R.; Hinchliffe, A. *Journal of Physical Organic Chemistry* **2001**, *14*, 709–715, DOI: 10.1002/poc.407.

(10) Irikura, K. K. *Journal of Physical and Chemical Reference Data* **2007**, *36*, 389–397, DOI: 10.1063/1.2436891.

(11) Bach, S.; Binder, A.; Montavon, G.; Klauschen, F.; Müller, K.-R.; Samek, W. *PLOS ONE* **2015**, *10*, 1–46, DOI: 10.1371/journal.pone.0130140.

(12) Binder, A.; Bach, S.; Montavon, G.; Müller, K.-R.; Samek, W. In *Information Science and Applications (ICISA) 2016*, Kim, K. J., Joukov, N., Eds.; Springer Singapore: Singapore, 2016, pp 913–922, DOI: 10.1007/978-981-10-0557-2_87.

(13) Lapuschkin, S.; Binder, A.; Montavon, G.; Müller, K.-R.; Samek, W. *Journal of Machine Learning Research* **2016**, *17*, 1–5.

(14) Hammond, J. R.; Kowalski, K. *The Journal of Chemical Physics* **2009**, *130*, 194108, DOI: 10.1063/1.3134744.

(15) Vincent, P.; Larochelle, H.; Bengio, Y.; Manzagol, P.-A. In *Proceedings of the Twenty-fifth International Conference on Machine Learning (ICML'08)*, ed. by Cohen, W. W.; McCallum, A.; Roweis, S. T., ACM: 2008, pp 1096–1103.

(16) Sculley, D.; Holt, G.; Golovin, D.; Davydov, E.; Phillips, T.; Ebner, D.; Chaudhary, V.; Young, M. In *SE4ML: Software Engineering for Machine Learning (NIPS 2014 Workshop)*, 2014.

(17) Thomsen, J.; Meyer, B. *Journal of Magnetic Resonance (1969)* **1989**, *84*, 212–217, DOI: 10.1016/0022-2364(89)90021-8.

(18) Baskin, I. I.; Winkler, D.; Tetko, I. V. *Expert Opinion on Drug Discovery* **2016**, *11*, PMID: 27295548, 785–795, DOI: 10.1080/17460441.2016.1201262.

(19) Kearnes, S.; McCloskey, K.; Berndl, M.; Pande, V.; Riley, P. *Journal of Computer-Aided Molecular Design* **2016**, *30*, 595–608, DOI: 10.1007/s10822-016-9938-8.

(20) Ma, J.; Sheridan, R. P.; Liaw, A.; Dahl, G. E.; Svetnik, V. *Journal of Chemical Information and Modeling* **2015**, *55*, PMID: 25635324, 263–274, DOI: 10.1021/ci500747n.

(21) Ramakrishnan, R.; Anatole von Lilienfeld, O. *ArXiv e-prints* **2015**.

(22) Wei, J. N.; Duvenaud, D.; Aspuru-Guzik, A. *ACS Central Science* **2016**, *2*, PMID: 27800555, 725–732, DOI: 10.1021/acscentsci.6b00219.

(23) Hase, F.; Valleau, S.; Pyzer-Knapp, E.; Aspuru-Guzik, A. *Chem. Sci.* **2016**, *7*, 5139–5147, DOI: 10.1039/C5SC04786B.

(24) Gastegger, M.; Behler, J.; Marquetand, P. *ArXiv e-prints* **2017**.

(25) Bojarski, M.; Del Testa, D.; Dworakowski, D.; Firner, B.; Flepp, B.; Goyal, P.; Jackel, L. D.; Monfort, M.; Muller, U.; Zhang, J.; Zhang, X.; Zhao, J.; Zieba, K. *ArXiv e-prints* **2016**.
(26) Finnegan, A. I.; Song, J. S. *bioRxiv* **2017**, DOI: 10.1101/105957.
(27) Lundberg, S.; Lee, S.-I. *ArXiv e-prints* **2016**.
(28) https://en.wikipedia.org/wiki/Decision_boundary (accessed 05/20/2017).
(29) Nakata, M.; Shimazaki, T. *Journal of Chemical Information and Modeling*, *0*, PMID: 28481528, null, DOI: 10.1021/acs.jcim.7b00083.
(30) Yao, K.; Herr, J. E.; Parkhill, J. *The Journal of Chemical Physics* **2017**, *146*, 014106, DOI: 10.1063/1.4973380.
(31) Li, Y.; Tarlow, D.; Brockschmidt, M.; Zemel, R. *ArXiv e-prints* **2015**.
(32) Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; Dahl, G. E. *ArXiv e-prints* **2017**.
(33) Kipf, T. N.; Welling, M. *arXiv preprint arXiv:1609.02907* **2016**.
(34) Ruddigkeit, L.; van Deursen, R.; Blum, L. C.; Reymond, J.-L. *Journal of Chemical Information and Modeling* **2012**, *52*, PMID: 23088335, 2864–2875, DOI: 10.1021/ci300415d.
(35) RDKit: Open-source cheminformatics., http://www.rdkit.org (accessed 05/07/2017).
(36) http://jupyter.org/ (accessed 06/05/2017).
(37) Pérez, F.; Granger, B. E. *Computing in Science and Engineering* **2007**, *9*, 21–29, DOI: 10.1109/MCSE.2007.53.
(38) Lapuschkin, S., https://github.com/sebastian-lapuschkin/lrp_toolbox (accessed 05/20/2017).
(39) Srinivasan, V., https://github.com/VigneshSrinivasan10/interprettensor (accessed 05/20/2017).
(40) Shiebler, D., https://github.com/dshieble/Tensorflow_Deep_Taylor_LRP (accessed 05/20/2017).
(41) Aidas, K.; Angeli, C.; Bak, K. L.; Bakken, V.; Bast, R.; Boman, L.; Christiansen, O.; Cimiraglia, R.; Coriani, S.; Dahle, P.; Dalskov, E. K.; Ekström, U.; Enevoldsen, T.; Eriksen, J. J.; Ettenhuber, P.; Fernández, B.; Ferrighi, L.; Fliegl, H.; Frediani, L.; Hald, K.; Halkier, A.; Hättig, C.; Heiberg, H.; Helgaker, T.; Hennum, A. C.; Hettema, H.; Hjertenæs, E.; Høst, S.; Høyvik, I.-M.; Iozzi, M. F.; Jansík, B.; Aa. Jensen, H. J. Aa.; Jonsson, D.; Jørgensen, P.; Kauczor, J.; Kirpekar, S.; Kjærgaard, T.; Klopper, W.; Knecht, S.; Kobayashi, R.; Koch, H.; Kongsted, J.; Krapp, A.; Kristensen, K.; Ligabue, A.; Lutnæs, O. B.; Melo, J. I.; Mikkelsen, K. V.; Myhre, R. H.; Neiss, C.; Nielsen, C. B.; Norman, P.; Olsen, J.; Olsen, J. M. H.; Osted, A.; Packer, M. J.; Pawlowski, F.; Pedersen, T. B.; Provasi, P. F.; Reine, S.; Rinkevicius, Z.; Ruden, T. A.; Ruud, K.; Rybkin, V. V.; Saek, P.; Samson, C. C. M.; de Merás, A. S.; Saue, T.; Sauer, S. P. A.; Schimmelpfennig, B.; Sneskov, K.; Steindal, A. H.; Sylvester-Hvid, K. O.; Taylor, P. R.; Teale, A. M.; Tellgren, E. I.; Tew, D. P.; Thorvaldsen, A. J.; Thøgersen, L.; Vahtras, O.; Watson, M. A.; Wilson, D. J. D.; Ziolkowski, M.; Ågren, H. *WIREs Comput. Mol. Sci.* **2014**, *4*, 269–284, DOI: 10.1002/wcms.1172.
(42) https://figshare.com/ (accessed 06/05/2017).
(43) Martn Abadi; Ashish Agarwal; Paul Barham; Eugene Brevdo; Zhifeng Chen; Craig Citro; Greg S. Corrado; Andy Davis; Jeffrey Dean; Matthieu Devin; Sanjay Ghemawat; Ian Goodfellow; Andrew Harp; Geoffrey Irving; Michael Isard; Jia, Y.; Rafal Jozefowicz; Lukasz Kaiser; Manjunath Kudlur; Josh Levenberg; Dan Mané; Rajat Monga; Sherry Moore; Derek Murray; Chris Olah; Mike Schuster; Jonathon Shlens; Benoit Steiner; Ilya Sutskever; Kunal Talwar; Paul Tucker; Vincent Vanhoucke; Vijay Vasudevan; Fernanda Viégas; Oriol Vinyals; Pete Warden; Martin Wattenberg; Martin Wicke; Yuan Yu; Xiaoqiang Zheng TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems., Software available from tensorflow.org, 2015.
(44) https://github.com/tensorflow/tensorflow (accessed 06/05/2017).
(45) Angeletti, G., https://gist.github.com/blackecho/3a6e4d512d3aa8aa6cf9 (accessed 06/05/2017).
(46) Shao, Y.; Gan, Z.; Epifanovsky, E.; Gilbert, A. T. B.; Wormit, M.; Kussmann, J.; Lange, A. W.; Behn, A.; Deng, J.; Feng, X.; Ghosh, D.; Goldey, M.; Horn, P. R.; Jacobson, L. D.; Kaliman, I.; Khaliullin, R. Z.; Kús, T.; Landau, A.; Liu, J.; Proynov, E. I.; Rhee, Y. M.; Richard, R. M.; Rohrdanz, M. A.; Steele, R. P.; Sundstrom, E. J.; Woodcock III, H. L.; Zimmerman, P. M.; Zuev, D.; Albrecht, B.; Alguire, E.; Austin, B.; Beran, G. J. O.; Bernard, Y. A.; Berquist, E.; Brandhorst, K.; Bravaya, K. B.; Brown, S. T.; Casanova, D.; Chang, C.-M.; Chen, Y.; Chien, S. H.; Closser, K. D.; Crittenden, D. L.; Diedenhofen, M.; DiStasio Jr., R. A.; Dop, H.; Dutoi, A. D.; Edgar, R. G.; Fatehi, S.; Fusti-Molnar, L.; Ghysels, A.; Golubeva-Zadorozhnaya, A.; Gomes, J.; Hanson-Heine, M. W. D.; Harbach, P. H. P.; Hauser, A. W.; Hohenstein, E. G.; Holden, Z. C.; Jagau, T.-C.; Ji, H.; Kaduk, B.; Khistyaev, K.;

Kim, J.; Kim, J.; King, R. A.; Klunzinger, P.; Kosenkov, D.; Kowalczyk, T.; Krauter, C. M.; Lao, K. U.; Laurent, A.; Lawler, K. V.; Levchenko, S. V.; Lin, C. Y.; Liu, F.; Livshits, E.; Lochan, R. C.; Luenser, A.; Manohar, P.; Manzer, S. F.; Mao, S.-P.; Mardirossian, N.; Marenich, A. V.; Maurer, S. A.; Mayhall, N. J.; Oana, C. M.; Olivares-Amaya, R.; O'Neill, D. P.; Parkhill, J. A.; Perrine, T. M.; Peverati, R.; Pieniazek, P. A.; Prociuk, A.; Rehn, D. R.; Rosta, E.; Russ, N. J.; Sergueev, N.; Sharada, S. M.; Sharmaa, S.; Small, D. W.; Sodt, A.; Stein, T.; Stück, D.; Su, Y.-C.; Thom, A. J. W.; Tsuchimochi, T.; Vogt, L.; Vydrov, O.; Wang, T.; Watson, M. A.; Wenzel, J.; White, A.; Williams, C. F.; Vanovschi, V.; Yeganeh, S.; Yost, S. R.; You, Z.-Q.; Zhang, I. Y.; Zhang, X.; Zhou, Y.; Brooks, B. R.; Chan, G. K. L.; Chipman, D. M.; Cramer, C. J.; Goddard III, W. A.; Gordon, M. S.; Hehre, W. J.; Klamt, A.; Schaefer III, H. F.; Schmidt, M. W.; Sherrill, C. D.; Truhlar, D. G.; Warshel, A.; Xua, X.; Aspuru-Guzik, A.; Baer, R.; Bell, A. T.; Besley, N. A.; Chai, J.-D.; Dreuw, A.; Dunietz, B. D.; Furlani, T. R.; Gwaltney, S. R.; Hsu, C.-P.; Jung, Y.; Kong, J.; Lambrecht, D. S.; Liang, W.; Ochsenfeld, C.; Rassolov, V. A.; Slipchenko, L. V.; Subotnik, J. E.; Van Voorhis, T.; Herbert, J. M.; Krylov, A. I.; Gill, P. M. W.; Head-Gordon, M. *Mol. Phys.* **2015**, *113*, 184–215.

(47) Turney, J. M.; Simmonett, A. C.; Parrish, R. M.; Hohenstein, E. G.; Evangelista, F. A.; Fermann, J. T.; Mintz, B. J.; Burns, L. A.; Wilke, J. J.; Abrams, M. L.; Russ, N. J.; Leininger, M. L.; Janssen, C. L.; Seidl, E. T.; Allen, W. D.; Schaefer, H. F.; King, R. A.; Valeev, E. F.; Sherrill, C. D.; Crawford, T. D. *Wiley Interdisciplinary Reviews: Computational Molecular Science* **2012**, *2*, 556–565, DOI: 10.1002/wcms.93.

(48) Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; Duchesnay, E. *Journal of Machine Learning Research* **2011**, *12*, 2825–2830.

(49) LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. In *Intelligent Signal Processing*, IEEE Press: 2001, pp 306–351.

List of Corrections