Machine Learning techniques for predicting molecular properties

Viviana Petrescu
I&C, EPFL

. *Abstract*—**The high computational cost of quantum chemistry calculations have prompted the use of less expensive machine learning methods for predicting molecular properties in chemical compound space. Finding good feature representations for molecules is hard, in part because of the graph-like structure geometry of the molecules that need to be represented as high dimensional vectors. Another difficulty in applying machine learning to molecular data is choosing and training the right models across a large dataset of labeled data. This write-up proposes suggestions to overcome some of the above issues.**

Index Terms—**molecules descriptors, GP-UCB, unlabeled data**

I. Introduction

The discovery of new molecular materials in chemistry has the potential of solving many of the problems we face today. Having a system which predicts both accurately and at a small computational cost the properties of new materials is highly desirable and has applications ranging from novel drugs discovery, water purification to efficient materials design for high energy transmission and storage [2].

Proposal submitted to committee: June 13th, 2009; Candidacy exam date: June 20th, 2009; Candidacy exam committee: Exam president, thesis director, co-examiner.

II. Background Work

A. Learning Invariant Representations of Molecules for Atomization Energy prediction

While domain specific descriptors for molecules exist [13], recent work [9] has proposed to predict properties of a molecule of size N only from the 3D positions of the atoms R_i $i \in 1..N$ and their nuclear charge Z_i $i \in 1..N$. This has prompted the introduction of the Coloumb Matrix descriptor, whose individual entries appear in the Schroedinger equation. It is defined by a NxN matrix with entries given by:

$$M(i,j) = \begin{cases} 0.5 * Z_i^{2.4} & if i = j \\ Z_i * Z_j / ||R_i - R_j|| & otherwise \end{cases} \tag{1}$$

where $||Ri - Rj||$ represents the distance between the atoms i and j.

The dimensionality of the Coloumb Matrix is given by the number of atoms in a molecule. Since that varies across molecules in a dataset, one common trick is to pad with 0's the matrices corresponding to small molecules until they reach the maximum molecule size in a dataset. This limits the size of the molecules that can be used, since the descriptor has complexity $O(N^2)$, where N is the number of atoms.

Desired properties of descriptor Due to the graph-like structure of the molecules, finding a fixed size representation is difficult. The desired properties of a molecular descriptor are invariance to translation and rotation of the molecule and invariance to the indexing of the atoms.

Solved using sorted or Random Coloumb While the Coloumb Matrix representations solves the rotation and translation invariance through the use of distances between atoms $||R_i - R_j||$, invariance in atom indexing still needs to be tackled since any permutation of atom indexes results in a valid Coloumb descriptor. Two variations of the descriptor are proposed in [5]. The first one, called Sorted Coloumb, uses the permutation of the atoms given by the sorting of the row norms of a valid Coloumb matrix. Any molecule has a unique representation given by the Sorted Coloumb matrix. The second representations is based on the idea that the norms of the rows can have very similar values in practice and the sorting, therefore also the atom indexing, is subject to small noise. The new descriptor, Random Coloumb Matrices is a collection of Sorted Coloumb matrices. For every molecule, approximatively 10 Randomly Sorted Coloumb matrices are drawn. One Random Sorted Coloumb Marix is obtained by sorting the set of rows according to their norm at which a small Gaussian noise was added. New predictions can be made by averging the prediction results for 10 such instantations.

The performane of the descriptors was evaluated on predicting the atomization energy on a dataset of 7165 molecules with at most 23 atoms per atom. The predicted values range from -800 to -2000 kcal/mol. The splitting of the data into 5 folds for cross validation is done using stratified sampling. The molecules were clustered into 5 sets with similar atomization energy levels and the folds were created by randomly selecting one molecule from each bucket.

For experimental evaluation of the performance of descriptors, both kernel ridge regression and multy layer feed forwards networks were tried. The best test performance of 3.1 kcal/mol MAE was obtained using a neural network with Randomly Sorted Coloumb matrix. The improvement was three fold with respect to the previous state of the art, which gave a MAE of 9kcal/mol, while a performance level of 1kcal/mol is desired to reach chemical accuracy level. Later work [3] improved upon this result by reducing the current state-of-the-art to only 1.5kcal/mol using a bag of bonds descriptor with Laplacian kernel.

While recently Neural Networks (especially deep networks) have achieved state-of-the-art in many fields ranging from computer vision, natural language processing and speech recogition, one of the main challenges they pose is the difficulty in training for people without domain expertise. In order to obtain the competitive performance of 3.1kcal/mol using multi layer feed forward neural networks, multiple tricks have been used, inspired from [6]. First of all, taking the real entries of the Coloumb matrix as input to the neural net proved to perform poorly. To overcome this, a binarization step was performed which added an extra dimensionality to the dataset. Specifically, every $x \in R^D$ representing a flatten Coloumb matrix is mapped to a new binarized descriptor $y \in R^{DxM}$ such that $y_i \in R^{1xM}$ is computed by taking shifted versions of x_i entry that are passed through a sigmoidal function

$$y_i = [..., tanh(\frac{x_i - \theta}{\theta}), tanh(\frac{x_i}{\theta}), tanh(\frac{x_i + \theta}{\theta}), ..] \quad (2)$$

Depending on the range of x_i, the size of y_i can vary acrosss dimensions if we ignore the saturated values. The other parameters that are selected using cross validation are the number of hidden units per layer, the number of layers, the learning rate.

B. Self-Taught Learning: Transfer Learning from Unlabeled Data

One challenge in general supervised learning tasks is to obtained labeled data. Usually, the higher the dimensionality of the input representations, the more labeled data is required for good generalization performance. Since obtaining large amounts of labeled data can require tedious manual labor work, people often employ Amazon Mechanical Turk for obtaining labeled data. To overcome this, this paper introduces another machine learning framework called *self-taught learning*, which has as core idea the fact that unlabeled dataset is much easier to obtain. Therefore, in the new setup, both labeled and unlabeled data is used for training a classifier. In

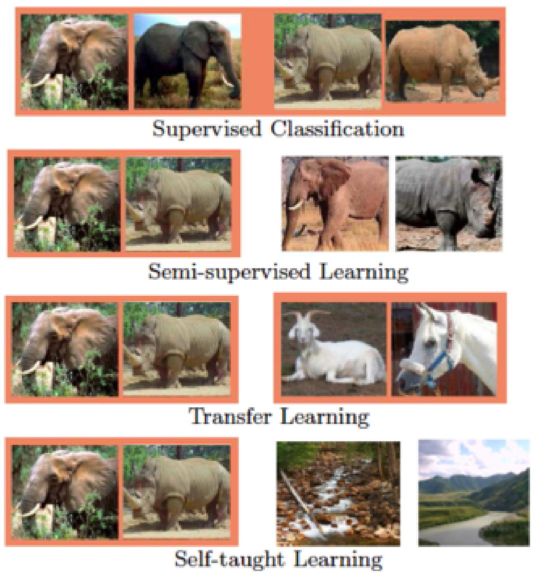

Fig. 1. Different learning setups

Self-Taught learning, labeled data is transformed into a higher level representation learned through the use of a large number of unlabeled data. The new representation is used for training a classifier on the labeled data. This setup is envisioned to have a bigger advantage when the available labeled data is scarce.

According to the type of data available (labeled and/or unlabeled) there are a number of training/testing setups currently used:

- Supervised Classification - training labels are provided
- Semi-Supervised Classification - training labels are not provided, but test data comes from the same class distribution as the training data
- Transfer Learning - training labels are provided but the test data is assumed to be from a different distribution
- Self-Taught Learning - training data does not have labels and it is also not assumed to come from the same distribution as the test data

An intuitive example is shown in Fig 1, where the orange bounded boxes represent labeled data.

1) Problem formulation: Self taught learning can be expressed as a two step optimization problem.

In the first step, the unlabeled data $x_u^{(i)} \in R^n, i = 1..k$ is used to find a set of *bases* $b_i \in R^n, i = 1..s$ and a set of *sparse activations* $a_j^{(i)}$ with $a^{(i)} \in R^s$ such that $x^{(i)}$ can be reconstructed from the s bases vectors b and its corresponding activation weights $a^{(i)} \in R^s$ given by the formula:

$$\begin{array}{c} \min_{a,b} \sum_{i=1}^{k} ||x_u^{(i)} - \sum_j a_j^{(i)} b_j||_2^2 + \beta ||a^{(i)}||_1 \\ s.t. ||b_j||_2 \leq 1, j = 1..s \end{array} \quad (3)$$

The equation above can be easily optimized using the algorithm from [4] by alternating the optimization in a and b, e.g. fix one variable and optimize the other one at every step. The problem is equivalent with a least square L1-regularized problem in variable a and a least square constrained problem in variable b. The L1-regularization enforces sparse representations and it was prefered to other forms of regulatization because experimentally appeared to perform better across different datasets. We note that PCA can also be used for finding a new representation of the input, but it leads to different solutions for which the activations are linear combination of the input. Moreover, PCA can not be used for generating more bases than the input dimension.

In the second step, the bases b learned in the previous step are used for finding a higher level representation for labeled input data $x_l^{(i)}, i = 1..m$ with corresponding target values $y_l^{(i)}, i = 1..m$. The new representation given by the set of activation vectors $\hat{a}(x_l^{(i)}), i = 1..m$ is obtained by solving another L1- regularized optimization problem:

$$\hat{a}(x_l^{(i)}) = \arg\min_{a^{(i)}} ||x_l^{(i)} - \sum_j a_j^{(i)} b_j||_2^2 + \beta||a^{(i)}||_1. \quad (4)$$

As a last step, a classifier is trained on the pairs $\{\hat{a}(x_l^{(i)}), y^{(i)}\}, i = 1..m$ and compared with a classifier (same type e.g. SVM) trained on the initial pairs $\{x_l^{(i)}, y^{(i)}\} i = 1..m$.

2) Experimental Results: Self taught learning was extensively tested across a range of different applications such as computer vision with raw pixel intensities as features, natural language processing with bag of words features and speech recognition with frequency histograms. Using an SVM as the classifier with sparse activation features as input instead of the initial raw features performed better in all except one tasks. This suggests that such a problem setup can be employed to datasets belonging to different domains besides the ones presented here.

C. *Information-Theoretic Regret Bounds for Gaussian Process Optimization in the Bandit Setting*

PUT ALSO PICTURE WITH EIGENSPECTRUM DECAY

Optimizing a black box function $f : D- > R$ that is expensive to evaluate is a common problem with applications ranging from active user modelling, hierarchical reinforcement learning[1] and more recently, hyper parameter optimization of machine learning models [11]. This problem can be posed as a multi-armed bandit problem where the function to be estimated is either sampled from a Gaussian Process or has low norm in reproducing Kernel Hilbert space.

The optimization is done in a sequential manner, where at round t a point x_t in the domain is chosen according to an *acquisition function* and the value $f(x_t)$ is evaluated. The goal is to maximize f with as few samples as possible. Among the common used acquistion functions we note the Probabilty of Improvement, Maximum Expected Improvement or Upper Confidence Bound (called GP-UCB, the focus of this paper), with the latter two experimentally shown to perform better in practice.

The effectiveness of a sampling scheme in finding the optimal $x* = \arg\max_{x\in D} f(x)$ ($x*$ may not be unique) is evaluated using the *instantaneous* and the *cumulative regret*. The instantenous regret $r_t = f(x*) - f(x_t)$ is the error that we make at round t for choosing point x_t instead of $x*$. The cumulative regret is the error accumulated up to the current round $R_T = \sum_{t=1..T} r_t$.

Besides the practical success of GP optimization, few was known in practice about their convergence properties for finite and infinte input dimensional space. The main contribution of the paper is to show sublinear convergence rates of the cumulative regret for GP-UCB in T, the number of rounds. In other words, as T goes to infinity, we expect the cumulative regret to be 0 $lim \frac{R_T}{T} = 0$.

The bound on the cumulative regret R_T is obtained in two steps. First bound R_T using the information gain, then the information gain is bound using the empirical spectrum operator and the kernel operator spectrum.

For simplicity and since RKHS is only relevent in a non-Bayesian setting, its details will be skipped here.

1) Gaussian Processes basics: A Gaussian Process (GP) defines a distribution over functions f, $f \sim GP(u(x), k(x, x'))$, in a similar way in which probability densities define distributions over random variables. It is fully specified by its mean $u(x)$ and its covariance matrix K. In a Bayesian setting, we assume our function f is sampled from a Gaussian with prior $GP(0, k(x, x'))$, where k is the kernel or covariance matrix. The most common kernel types are the linear, squared exponential and Matern kernels.

For T sampled points in the presence of noise, we have $y_t = f(x_t) + \epsilon_t$, where $\epsilon_t \in N(0, \sigma^2)$. The posterior over f, conditioned on $y_{1..T}$ and $x_{1..T}$ is given by $P(f_T|y_{1:T}, x_{1..T}) = N(u_T(x), \sigma_T^2(x))$, where:

- K_T is positive definite with entries $k(x, x'), x, x' \in A_T$
- $k_T(x) = [k(x_1, x), ...k(x_T, x)]^T$
- $k_T(x, x') = k(x, x') - k_T(x)^T (K_T + \sigma^2 I)^{-1} 1 k_T(x')$
- $u_T(x) = k_T(x)^T (K_T + \sigma^2 I)^{-1} y_T$
- $\sigma_T^2(x) = k_T(x, x)$

2) Information Gain and Experimental Design: Unlike function optimization where the goal is to find the maximum of a function, in experimental design the goal is to find a good approximation of the function globally with few samples as possible. We note that the same algorithm can not be employed for both tasks, since in function optimization we might want to avoid sampling in the regions where we are confident that the function has small values.

For a sampled set $x \in A$, we define $y_A = f_A + \epsilon_A$, where $f_A = [f(x)]_{x\in A}$ and the noise $\epsilon_A \sim N(0, \sigma^2)$. The mutual information gain is

$$I(y_A; f) = H(y_A) - H(y_A|f) \quad (5)$$

where $H(y_A)$ and $H(y_A|f)$ represent the entropy or uncertainty in y_A and the uncertainty in y_A if we know f. In other words, information gain expresses the reduction in uncertainy about f given y_A.Shouldn t be the other way around?

Using the entropy formula for a Gaussian $H(N(u, \sigma)) = \frac{1}{2} log|2\pi e \Sigma|$ we can rewrite $F(A) = I(y_A; f) = I(y_A; f_A) = \frac{1}{2} log|I + \sigma^{-2} K_A|$ Finding the subset A in the domain D with $|A| \leq |T|$ that maximizes $I(y_A; f)$ is NP-hard. In practice, greedy approximation solutions are used that find a near-optimal solution by choosing a sequence of points in A with maximum variance at every step. Thus, the following

Kernel	Linear	RBF	Matern
Regret R_T	$d\sqrt{T}$	$\sqrt{T(log(T)^{d+1})}$	$v + d(d+1)/2v + d(d+1)$

TABLE I
REGRET BOUNDS.

equivalence relation holds at round t:

$$x_t = \arg\max_{x \in D} F(A_{t-1} \cup x) \iff x_t = \arg\max_{x \in D} \sigma_{t-1}(x) \quad (6)$$

with $t \in 1...T$

In [7] it is shown that for any submodular function $F(A)$ and set A_T obtained using the greedy aproach from above, it holds that:

$$F(A_T) \geq (1 - \frac{1}{e}) \max_{|A| \leq T}(F(A)) \quad (7)$$

Thus, a greedy set creation of A_T leads to a solution close to the optimal.

3) GP-UCB: For the function optimization problem, a good sampling sequence x_t is one that makes a tradeoff between exploration and exploitation of the search space. By choosing points with high variance (exploitation) we try to reduce the overall uncertainty. By choosing points with high mean (exploitation) we try to concentrate samples in a region were we are more confident that is close to the optimal. This idea is expressed by choosing in every round t, a point x_t according to an acquistion function $x_t = \arg\max_{x \in D} u_{t-1} + \beta_t^{1/2} \sigma_{t-1}(x)$ with β_t depending on the kernel type. The function that is being maximized is also called the acquistion function or utility function and corresponds to Upper Confidence Bound criterion.

To sum up, at every round, the GP-UCB algorithm selects a point according to the above equation, evaluates $y_t = f(x_t) + \epsilon$ and updates u_t and σ_t using Bayes rule from equation 1.

An important property of the UCB defined as above is that it does not choose points x for which the upper confidence value is smaller than the maximum lower confidence value found so far. This can prune large subareas of the search space.

4) Regret Bounds: This paper is the first to prove sublinear convergence rates for the cumulative regret R_T with both finite and infinite input domain D with smoothness assumptions on the kernel. They show that in the limit GP-UCB has no regret $lim \frac{R_T}{T} = 0$ and the convergence results for the most popular kernels can be found in table I, up to polylog factors. We notice that the dependance on the dimensionality d of the input is weak. Namely, for the linear kernel the dependency is linear and for the squared exponential kernel d appears only as a power of $log(T)$.

5) Bounding the Regret using Information Gain: The first step of the proof is to bound R_T by the maximum information gain $\gamma_T = \max_{A \subseteq D, |A| = T} I(y_A; f_A)$ after T rounds. In particular, it is shown that with high probability

- $Pr(R_T \leq \sqrt{C_1 T \beta_T \gamma_T}) \leq 1 - \delta$ for finite D
- $Pr(R_T \leq \sqrt{C_2 T \beta_T \gamma_T} + 2) \leq 1 - \delta$ for D compact and convex

where $\delta \in (0, 1)$ and appropiate constants C_1, C_2, β_t . The second inequality holds by making further assumptions that the function is smooth, e.g. by enforcing that the probability of the partial derivatives of the function to have a large value is small.

6) Bounding the Information Gain using the empiricial spectrum: The case of infinite input dimensionionality are treated similarly with the finite case by assuming the existence of a discretized set $D_T \subset D, T \in N$ with nearest neighbor distance O(1), dense in the limit. As stated previously, the value of $\gamma_T = \max_{A \subset D_T, |A| = T} I(y_A; f_A)$ can be bounded by the greedy maximization value $\gamma_T \leq \frac{1}{1-e^{-1}} F(A_T)$ since it is a submodular function.

Choosing a point x_t is equivalent with selecting a vector $v_t \in R^{|D|}$ with values 0's except the entry at position t, thus $f \sim N(v_t f, \sigma^2)$. By relaxating the constraint to $||v_t|| = 1$, it is shown that the v_t's are selected among the eigenvectors of the kernel matrix $K_{D_T} = k(x, x')x, x' \in D_T$. The maximum information gain is further bounded by an expression involving the eigenvalues $\hat{\lambda_t}$ of K_{D_T} as follows:

$$\gamma_T \leq \frac{0.5}{1 - e^{-1}} \max_{m_1, m_2, ..., m_D} \sum_{t=1}^{|D|} log(1 + \sigma^{-2} m_t \hat{\lambda_t}) \quad (8)$$

where $m_t \in N$, $\sum_t m_t = T$ and $\hat{\lambda_1} \geq \hat{\lambda_2} \geq \hat{\lambda_3} \geq ...$ Here we have considered a worst case scenario of assigning eingevectors of K_T to v_t. In the next step, the information gain is bounded using the tail spectrum of K_{D_T}, $B(T_*) = \sum_{t=1}^{T_*} \hat{\lambda_t}$ and $n_T = \sum_{t=1}^{T_*} \hat{\lambda_t}$ for any $T_* = 1..T$:

$$\gamma_T \leq O(\sigma^{-2}[B(T_*)T + T_* log n_T T)]) \quad (9)$$

From the equation above, we conclude that if the tail spectrum $B(T_*)$ is small, or in others words, the eingenspetrum of K_{D_T} decays fast, the bound on γ_T is more tight.

7) Bound the empirical spectrum by the operator spectrum: For a kernel k with $k(x, x)u(x) = 1$ where u(x) is uniformly distributed on D, Mercer's theorem assures the existence of a discrete kernel eigenspectrum $\lambda_s(x), \phi_s(x)$ with $k(x, x') = \sum \lambda_s \phi_s(x) \phi_s(x')$

Another contribution(?) of this paper is to bound the empirical tails spectrum $B(T_*)$ by the kernel operator tails eigenspectrum $B_k(T_*) = \sum_{t=1}^{T_*} \lambda_t$. Analytical expression bounds on $B_k(T_*)$ for the most common kernels are given by Seeger [10].

8) Conclusions: The contribution of the paper is two fold. First, a novel connection is being made between experimental design and GP optimization, by bounding the cumulative regret by the maximum information gain. Secondly, for the first time sublinear regret bounds for GP-UCB optimization are shown with weak dependence on the dimensionality of the input. The bound is derived in two steps. First, the cumulative regret is bounded by the maximum information gain which in turn is bounded by the tails of the empirical eigenspectrum which in turn is bound by the tails of the kernel operator spectrum. The bound is tighter, the more rapidly the kernel spectrum decays.

Although convergence rates are proven for infinite input dimensional spaces as well, the complexity of the GP step that incorporates the inversion of the covariance matrix makesthe complexity of the algorithm cubic in the number of samples T, which makes it often inpractical for high dimensional input.

III. Research Proposal

We propose a new descriptor, Bag of Bonds Histogram (BoBH), that is more robust to atom indexing by avoiding the sorting step. Its size is not dependent on the number of atoms in the molecules but on the number of different atom types. Every bag in a BoBH descriptor is a histogram of quantized distances which encodes information for all pairs of a certain type. The size of the histogram bag is given by the maximum distance between two pairs of atoms (pertaining to that bag type). The quantization level of the distances varies between bag types and we experimentally found that a quantization level of 0.2 or 0.25 perform well in practice, but this can vary for different datasets.

Thus, for a given molecule, the BoBH descriptor will encode:

- for every different atom type present in the dataset, count how many times it appears in the molecule
- for every pair of atoms of certain type, count how many times the distance between them is in a given quantization interval

An example is given in Fig 2 for a molecule C_4H_2O with a quantization level of $q = 1$. In this example, first the bags containing H were concatenated, then the ones containing C and in the end the ones containing O, resulting in a descriptor of size 21. In general, the size of the descriptor is given by $NA + \sum_i \frac{N_A*(N_A+1)}{2} * \frac{D_{max}}{q}$, where N_A is the number of atom types and D_{max} is the maximum distance between a pair of atoms.

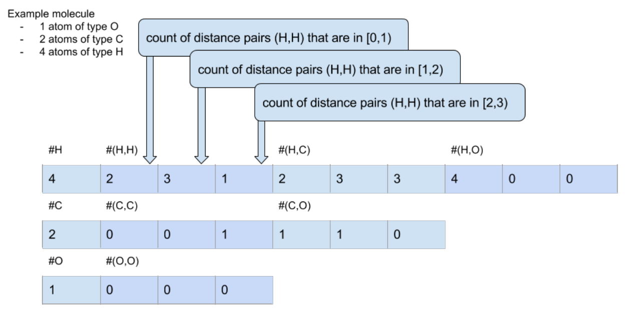

Fig. 2. Bag of Quantised Bonds. Sample computation of Bag of Bonds Histogram for C_2H_4O for a dataset which contains only atoms of type O, C and H. If the maximum distance between any two atoms in our dataset is 3, the size of the BoBH descriptor with quantization level 1 is 21, as shown above. The first entry encodes the number of H atoms, 4. The following 3 elements encode the number of (H,H) distances that fall in the interval [0,1), [1,2) and [2,3).

To sum up, all descriptors achieve translation and rotation invariance through the use of distances between atoms instead of the actual 3D positions. Sorted Coloumb and Randomly Sorted Coloumb achieve invariance to index permutation by sorting the rows of the matrix according to their norm. Bag of Bonds Histogram puts atom pairs with the same composition in the same bag and quantizes the distances into bins to bypass the sorting step present in other descriptors. Preliminary results on the same dataset as in [5] show promising results, obtaining a MAE of 2.6 kcal/mol.

Currently, preprocessing the molecular data to be used in a machine learning setting requires extensive domain specific knowledge. As an example, splitting the folds for cross validation using stratified sampling is a good strategy for obtaining good generalization performance but in practice, we should not expect to know the range of the test data. If we test the performance on a new molecule whose atomization energy is not within the bounds present in the training dataset, it is likely that the prediction will not be very accurate. Moreover, if we have a molecule with more atoms than any molecule present in the dataset, we would need to retrain our model with a descriptor of different size.

The same problem appears across chemical compound space if we try to predict the atomization energy of a molecule with the same number of atoms but with different atom types in its composition. In general, we do not have guarantees that a molecule with similar atom composition and similar atom type have target prediction in the same range as our training set.

The labeled datasets currently used (from *www.quantum-mechanics.org*) are in the order of 7k or 140k and are of relatively small size compared with datasets in other domains. Basically, they contain at most 6 atom types and at most 29 atoms per molecule. Moreover, in chemoinformatics the datasets can not be easily augumented by using external annotation tools since it requires domain knowledge and the labeling effort implies time intensive computations.

Although the self-taught learning problem in [8] was tested only for classification tasks and predicting properties of molecules is a regression task, employing large amounts of unlabeled molecular data for obtaining higher level representations of the molecules could lead to increased prediction performance.

In other words, our problem can be formulated as a semi-supervised, transfer-learning or self-taught learning problem by making it generalize across compound space and learn new embeddings of the atoms. As an example, learn the higher level features from one dataset containing $\{H, C, O, S, N\}$ and use this for a new set containg $\{H, C, O, S, Cl\}$.

As previously mentioned, a drawback of using neural networks is the difficulty in training. The authors of [5] make use of various tricks [6] to obtain state-of-the-art performance on the molecular datasets. A grid search approach for choosing the hyper parameters of a model leads to exponential time complexity in the number of parameters. Recent work [11], [12] suggests the use of Bayesian optimization techniques (such as GP-UCB or Bayesian neural networks) for hyper-parameters tuning. In this setup, the function to be maximized is the accuracy on the validation set while the input represents the concatentation of the hyper parameters (learning rate, type of activation function, number of units per hidden layer etc.). The speedup in training time can be quite significant, since techniques such as GP-UCB employ an exploration - exploitation trade-off scheme that prunes large areas of the search space. In [12] they experimentally demonstrate on a NLP task, that by using a simple model tuned in a Bayesian optimizatin setting can outperform the use of a more complex model (such as Recurrent Neural Networks). Such techniques which were sucessfully applied to NLP and vision tasks and can be employed to molecular datasets as well. One further direction is to compare the current existent models and descriptors tuned in a Bayesian optimization setting, and can search over a larger space of models to have a more robust overview of which model/descriptor performs better.

REFERENCES

[1] Eric Brochu, Vlad M. Cora, and O De Freitas. A tutorial on bayesian optimization of expensive cost functions, withapplicationtoactiveuser-modeling andhierarchical reinforcement learning. 2009.

[2] Johannes Hachmann, Roberto Olivares-Amaya, Sule Atahan-Evrenk, Carlos Amador-Bedolla, Roel S. Snchez-Carrera, Aryeh Gold-Parker, Leslie Vogt, Anna M. Brockway, and Aln Aspuru-Guzik. The harvard clean energy project: Large-scale computational screening and design of organic photovoltaics on the world community grid. *The Journal of Physical Chemistry Letters*, 2(17):2241–2251, 2011.

[3] O.A. von Lilienfeld K.-R. Mller K. Hansen, F. Biegler and A. Tkatchenko. *Interaction Potentials in Molecules and Non-Local Information in Chemical Space.* Phys. Rev. Lett. (March 10, 2014)., 2014.

[4] Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y. Ng. Efficient sparse coding algorithms. In *Advances in Neural Information Processing Systems 19*, pages 801–808. 2007.

[5] Grégoire Montavon, Katja Hansen, Siamac Fazli, Matthias Rupp, Franziska Biegler, Andreas Ziehe, Alexandre Tkatchenko, Anatole V Lilienfeld, and Klaus-Robert Müller. Learning invariant representations of molecules for atomization energy prediction. In *Advances in Neural Information Processing Systems*, pages 440–448, 2012.

[6] Grégoire Montavon, Genevieve B. Orr, and Klaus-Robert Müller, editors. *Neural Networks: Tricks of the Trade - Second Edition*, volume 7700 of *Lecture Notes in Computer Science*. Springer, 2012.

[7] G.L. Nemhauser, L.A. Wolsey, and M.L. Fisher. An analysis of approximations for maximizing submodular set functions–i. *Mathematical Programming*, 14(1):265–294, 1978.

[8] Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y. Ng. Self-taught learning: Transfer learning from unlabeled data. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 759–766, New York, NY, USA, 2007. ACM.

[9] Matthias Rupp, Alexandre Tkatchenko, Klaus-Robert Müller, and O. Anatole von Lilienfeld. Fast and accurate modeling of molecular atomization energies with machine learning. *Phys. Rev. Lett.*, 108:058301, Jan 2012.

[10] MW. Seeger, SM. Kakade, and DP. Foster. Information consistency of nonparametric gaussian process methods. *IEEE Transactions on Information Theory*, 54(5):2376–2382, May 2008.

[11] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. pages 2951–2959, 2012.

[12] Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Md, Prabhat, and Ryan P. Adams. Scalable Bayesian Optimization Using Deep Neural Networks, February 2015.

[13] Roberto Todeschini and Viviana Consonni. *Handbook of molecular descriptors*, volume 11. Wiley-VCH, 2000.