Learning over Molecules: Models

BY
Jian Li, Ethan Cowan, Weiyi Chen

Harvard University
Cambridge, Massachusetts
Feburary 2016

Course advisor: Finale Doshi-Velez Jian Li, Ethan Cowan, Weiyi Chen

Learning over Molecules: Models

Abstract

In this report, we tackle machine learning over molecular space by considering features that "identifies" molecules. We assess the viability of different machine learning algorithms by training a regressor model to predict energy values. In particular, we look several class of models including ridge regression, lasso regression, elastic net, neural network, ensemble methods, support vector machine and Gaussian process regression, whereby the prediction algorithm relies on a feature called Morgan fingerprints and its similarity measure between training data. On the given Kaggle training data and testing data, we find a random forest regression to be the most accurate for predicting HOMO-LUMO energy gap values.

Contents

1 Introduction

1.1 Background

According to the practical description [1], Solar power is one of the most promising technologies for renewable energy to reduce our dependence on fossil fuels. Unfortunately, most modern solar cells are based on silicon. These materials are rigid, expensive, and difficult to manufacture. On the other hand, *carbon* based solar cells could be cheap to produce, flexible, transparent, and be made and molded as easily as plastics. There's just one catch: no known organic photovoltaic molecules are as efficient as their silicon counterparts.

The Harvard Clean Energy Project has been using massive scales of computation to explore new possibilities for organic photovoltaics. The project uses density functional theory (DFT) to estimate the the properties of the molecules that determine their potential efficiency as solar cells. The main

quantity of interest is the difference in energy between the highest occupied molecular orbital (HOMO) and the lowest unoccupied molecular orbital (LUMO). It can take hours or days to compute this accurately on a modern computer.

1.2 Problem

Recently, it has become clear that machine learning might have something to say about this. It may be possible to sidestep these expensive DFT computations by learning a function from a feature representation of the molecule to the HOMO-LUMO gap.

In this report, we consider it as a regression problem: take molecular features and produce a real-valued prediction of what the DFT would calculate. Our purpose is to find better machine learning models for this problem could lead to new kind of materials and more efficient solar cells!

1.3 Kaggle requirement

We have one million molecules to train on, and are tasked with making predictions on another 800,000 or so. We are required to upload predictions to Kaggle, where a subset will be used to produce a "public leaderboard" and another subset will be used to reveal the final rankings at the end of the contest. Inside the data, we have access to complete information about the molecular structure in the form of a SMILES string. This is a representation that chemists like to use to encode molecular structures, e.g.

$$c1sc(-c2sc(-c3ccc(cc3) - c3scc4sccc34)c3[se]ccc23)c2sccc12$$

One of the challenges of making predictions about molecules is finding a good feature representation. Clearly, the SMILES string itself isn't going to be all that helpful. Luckily, many chemists have built tools to build interesting

representations. One such tool is a Python package called RDKit. With it we loaded SMILES strings and produce Morgan fingerprints as our features. Alternatively during our own testing, we use the features given from Kaggle, which produced 256-dimensional binary vectors.

2

Related Work

2.1 Neural network predictions

In 2012, Montavon et al. [6] have shown Coulomb matrices to be a useful representation for energetic predictions using neural networks. In particular, they propose the idea of random sampling of Coulomb matrices over the possible permutations of atomic indexing. Their best neural network predicted atomization energies significantly better than various kernel methods, but neural networks are diffcult and time-consuming to train. Furthermore, they show that kernel methods are less affected by the specific representation of the Coulomb matrix.

2.2 Representations

In 2014, Sun [10] has examined representations and kernels for machine learning over molecules. He considered cheminformatic feature vectors, graph based matrix representations, and molecular fingerprints. He presented the test accuracy of the representations and kernels used as well as a mean predictor reference. Despite the richness of the Coulomb matrix representation, it performs poorly on this subset of data. Molecular fingerprinting results in the smallest error.

Along with representations, Sun [10] also considered a simple RBF kernel, a random walk graph kernel, and a fingerprint similarity index. On a subset of CEP data, molecular fingerprinting predicted HOMO-LUMO gaps with the lowest error.

Our project is extended from Sun's, and explore class of machine learning algorithms based on his accomplishment.

2.3 Fingerprinting and Molecular Similarity

The RDKit suggested by TF has a variety of built-in functionality for generating molecular fingerprints and using them to calculate molecular similarity. Morgan fingerprints, as one family of fingerprints, better known as circular fingerprints [8], is built by applying the Morgan algorithm to a set of user-supplied atom invariants.

Information is available about the atoms that contribute to particular bits in the Morgan fingerprint via the bitInfo argument. In RDKit, the dictionary provided is populated with one entry per bit set in the fingerprint, the keys are the bit ids, the values are lists of (atom index, radius) tuples.

We borrow the conclusion of Sun's to use molecular fingerprinting, to predict HOMO-LUMO gaps, and will use Mogan fingerprinting 2048-bit vector data as our starting feature. But considering the huge data would be time-consuming, we will use 256-bit vector data given during experiment process.

3 Methods

3.1 Data

We use the data provided by Kaggle, originally from The Harvard Clean Energy Project, an initiative at Harvard University to identify organic molecules with promising photovoltaic properties. The entire data set features over 2 million molecules with energetic properties calculated through crowd-sourced quantum computations. These molecules are given in string representation called the Simplified Molecular-Input Line-Entry System (SMILES). This is one of the industry standards for molecular representation. The response variable we attempt to predict is the difference in energy between the highest occupied molecular orbital (HOMO) and lowest unoccupied molecular orbital (LUMO). This HOMO-LUMO gap energy can be used as a proxy for the photovoltaic efficiency of a molecule.

3.2 Representations

Working with molecular SMILES directly is difficult since regression is usually tailored to a vector of predictor variables, rather than strings. We consider using Morgan fingerprints representation, a 2048-bit vector which is more amenable to machine learning methods.

As described in section 2.2, Sun [10] uses a fingerprinting method that accounts for substructures in molecules. Here we use a similar path-based fingerprint implemented in RDKit [5]. This method finds all atomic chains up to length 7 in a molecule, accounting for bond type, order, and cycles. Each canonical fragment is hashed to set 2048 bit vector. Thus a molecular fingerprint indicates the presence or absence of substructures within a molecule. In addition to fixed-size fragments, user-specified substructures can be used, allowing for input of prior knowledge. Such fingerprinting methods have been widely used for comparing molecules in medicinal chemistry.

3.3 Models

Though obviously given the inherently nonlinear interactions governing molecular systems, we still use both linear regressors and non-linear regressors as the regression methods. At Sun's [10] thesis, a Gaussian process acted as a nonlinear interpolater to data, modeling some smooth underlying function [2]. Here we will explore more different class of learning algorithms. In this section, we will give a brief mathematical overview.

3.3.1 Ridge regression

Ridge regression addresses some of the problems of Ordinary Least Squares by imposing a penalty on the size of coefficients. The ridge coefficients minimize a penalized residual sum of squares,

$$\min_{w} ||Xw - y||_2^2 + \alpha ||w||_2^2$$

Here, $a \geq 0$ is a complexity parameter that controls the amount of shrinkage: the larger the value of a, the greater the amount of shrinkage and thus the coefficients become more robust to collinearity. [7]

3.3.2 Lasso regression

The Lasso is a linear model that estimates sparse coefficients. It is useful in some contexts due to its tendency to prefer solutions with fewer parameter values, effectively reducing the number of variables upon which the given solution is dependent. For this reason, the Lasso and its variants are fundamental to the field of compressed sensing. Under certain conditions, it can recover the exact set of non-zero weights.

Mathematically, it consists of a linear model trained with ℓ_1 prior as regularizer. The objective function to minimize is:

$$\min_{w} \frac{1}{2n_{samples}} ||Xw - y||_2^2 + a||w||_1$$

The lasso estimate thus solves the minimization of the least-squares penalty with $a||w||_1$ added, where a is a constant and $||w||_1$ is the ℓ_1-norm of the parameter vector. [4]

3.3.3 Elastic net

ElasticNet is a linear regression model trained with L1 and L2 prior as regularizer. This combination allows for learning a sparse model where few of the weights are non-zero like Lasso, while still maintaining the regularization properties of Ridge. We control the convex combination of L1 and L2 using the $l1_{ratio}$ parameter.

Elastic-net is useful when there are multiple features which are correlated with one another. Lasso is likely to pick one of these at random, while elastic-net is likely to pick both.

A practical advantage of trading-off between Lasso and Ridge is it allows Elastic-Net to inherit some of Ridge's stability under rotation.

The objective function to minimize is in this case

$$\min_{w} \frac{1}{2n_{samples}} ||Xw - y||_2^2 + a\rho||w||_1 + \frac{a(1-\rho)}{2}||w||_2^2$$

from Zou's introduction [12].

3.3.4 Neural Network

This jumps out the range of generalized linear regression. The most frequently used algorithm in neural network is Multi-layer Perception (MLP).

MLP is a supervised learning algorithm that learns a function $f(\cdot) : R^m \rightarrow R^o$ by training on a dataset, where m is the number of dimensions for input and o is the the number of dimensions for output. Given a set of features $X = x_1, x_2, \ldots, x_m$ and a target y, it can learn a non-linear function approximator for either classification or regression. It is different from logistic regression, in that between the input and the output layer, there can be one or more non-linear layers, called hidden layers.

The input layer consists of a set of neurons $\{x_i | x_1, x_2, \ldots, x_m\}$ representing the input features. Each neuron in the hidden layer transforms the values from the previous layer with a weighted linear summation$w_1x_1 + w_2x_2 + \ldots + w_mx_m$, followed by a non-linear activation function $g(\cdot) : R \rightarrow R$ - like the hyperbolic tan function. The output layer receives the values from the last hidden layer and transforms them into output values. [9]

3.3.5 Ensemble Methods

The goal of ensemble methods is to combine the predictions of several base estimators built with a given learning algorithm in order to improve generalizability / robustness over a single estimator. We use random forest as a representation of this class of methods.

In random forests, each tree in the ensemble is built from a sample drawn with replacement (i.e., a bootstrap sample) from the training set. In addition, when

splitting a node during the construction of the tree, the split that is chosen is no longer the best split among all features. Instead, the split that is picked is the best split among a random subset of the features. As a result of this randomness, the bias of the forest usually slightly increases (with respect to the bias of a single non-random tree) but, due to averaging, its variance also decreases, usually more than compensating for the increase in bias, hence yielding an overall better model. [3]

3.3.6 Support Vector Regression

a support vector machine constructs a hyperplane or set of hyperplanes in a high- or infinite-dimensional space, which can be used for regression. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training-data point of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier [11].

In simpliest case of SVM, we are given a training dataset of n points of the form

$$(\vec{x}_1, y_1), \ \ldots, \ (\vec{x}_n, y_n)$$

where the y_i are either 1 or 1, each indicating the class to which the point $\vec{x}_i$ belongs. Each $\vec{x}_i$ is a p-dimensional real vector. We want to find the "maximum-margin hyperplane" that divides the group of points $\vec{x}_i$ for which $y_i = 1$ from the group of points for which $y_i = -1$, which is defined so that the distance between the hyperplane and the nearest point $\vec{x}_i$ from either group is maximized.

Any hyperplane can be written as the set of points $\vec{x}$ satisfying

$$\vec{w} \cdot \vec{x} - b = 0$$

.

3.4 Experimental process

There are three levels in our experiment.

At the lowest level, we use a subset of one million molecules with 256-bit vector from the Clean Energy Project data set for training, pick randomly into 10,000 for training and 8000 for testing.

At the second level, we use the whole one million data set of provided 256-bit vector data to train the regressor, predict the given 800,000 test data, and submit to see our scores.

At the final level, once we confirm the method is one of the best algorithms for prediction with lowest error, we generated 2048-bit Morgan fingerprints for both one million training data and 800,000 testing data, using them to train and predict respectively, and finally submit as our formal submission.

The reason of this is that the higher the level, the more time-consuming of the program run, so we try avoid wasting at the higher level by observing the score difference at lower levels.

4

Results

Firstly, we present the test accuracy of the second level experiment, including data and methods used as well as the linear regression sample as a reference. Linear methods performs poorly on this kind of data. Ensemble methods (random forest and gradient boosting) and decision trees (decision tree and extra tree) result in the smallest error, while linear regression methods are the worst.

Method	Score (error)
Linear regression (sample)	0.29846
Lasso regression	0.40682
Adaboost regression	0.32119
Random forest (sample)	0.27207
Random forest (only log2 features selected)	0.27208
Random forest (only sqrt2 features selected)	0.27208
Gradient Boosting	0.28592
Decision Tree	0.27206
Extra Tree	0.27206
SVM (only trained with 100,000 data randomly picked)	0.28823
SVM (only trained with 10,000 data randomly picked)	0.29691

After first level experiments, we are almost sure linear model does not play well with this inherently nonlinear interactions governing molecular systems. We would not consider linear methods any more when using Morgan fingerprints. Note SVM is only trained with subset of data because it is too time-consuming if trained with whole dataset, and training with more data does not improve the score much, we can assume SVM does not convergence in this case and will no longer use it at higher level either.

In addition, third level with 2048-bit data has result as follows. Random forest results in the smallest error still. Note some of the cases are labeled 1024-bit, Morgan fingerprints can be generated as either 1024-bit or 2048-bit, we played with 1024-bit initially and converted to 2048-bit latter, but will present them altogether here. We also put a ridge regression performance here as a benchmark.

Method	Score (error)
Ridge regression (1024-bit)	0.14337
Random forest (1024-bit)	0.6446
Decision Tree (1024-bit)	0.08563
Random forest (2048-bit)	0.06148
Gradient Boosting (2048-bit)	0.17057

We held the first place on leaderboard with 0.6148 until the last 5 days beat by other teams. Then we continue to try different ways to optimize, but our methods are restricted in tree / ensemble methods, below are our further results. There are three main thoughts during our optimizations

- Data cleaning, remove useless data (all zeros in the same entry of all molecules)
- Parameters adjustment, select features based on feature importance given by random forest regressor
- Data combination, find out more features from RDKit and combine together to acquire additional infomation

Method	**Score (error)**
Random forest (parameters optimized)	0.06004
Random forest (new fingerprinting, MACCS)	0.23402
Random forest (combined fingerprints of MACCS and Morgan)	0.06013
Extra tree (945-bit, selected important features)	0.05899

5 Conclusions

In this paper we have examined different classes of methods for machine learning over molecules. We considered ridge regression, lasso regression, elastic net, neural network, ensemble methods and support vector regressions. Along with these methods, we considered different ways to optimize our result restricting in ensemble / tree methods based on our previous progress, including data exploration to get more data, feature engineering to extract useful data, parameters adjustment to customize methods to the data.

Based on related work, we utilize the conclusion of molecular fingerprinting predicted HOMO-LUMO gaps with the lowest error. Though the random forest method we used did not prove to be the best teams among top 3, also we had programming problem when trying to utilize Gaussian process to process our data as submission ideas [1] suggested, the primary goal of this practical was exploratory, we have established a foundation for further work and started to

know how to play data with machine learning.

References

[1] Predict the efficiency of novel organic photovoltaic molecules description. https://inclass.kaggle.com/c/cs181-s16-practical-1.

[2] Albert P Bartók, Mike C Payne, Risi Kondor, and Gábor Csányi. Gaussian approximation potentials: The accuracy of quantum mechanics, without the electrons. *Physical review letters*, 104(13):136403, 2010.

[3] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[4] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.

[5] Greg Landrum. Rdkit: Open-source cheminformatics. *Online). http://www. rdkit. org. Accessed*, 3(04):2012, 2006.

[6] 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.

[7] Ryan M Rifkin and Ross A Lippert. Notes on regularized least squares. 2007.

[8] David Rogers and Mathew Hahn. Extended-connectivity fingerprints. *Journal of chemical information and modeling*, 50(5):742–754, 2010.

[9] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

[10] Hong Yang Sun. *Learning over Molecules: Representations and Kernels*. PhD thesis, 2014.

[11] Johan AK Suykens and Joos Vandewalle. Least squares support vector machine classifiers. *Neural processing letters*, 9(3):293–300, 1999.

[12] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.

Colophon

This report was typeset using LaTeX, originally developed by Leslie Lamport and based on Donald Knuth's TeX. The body text is set in 11 point Arno Pro, designed by Robert Slimbach in the style of book types from the Aldine Press in Venice, and issued by Adobe in 2007. A template, which can be used to format a PhD thesis with this look and feel, has been released under the permissive MIT (X11) license, and can be found online at github.com/suchow/ or from the author at suchow@post.harvard.edu.