Machine learning techniques for predicting molecular properties

Viviana Petrescu

Overview

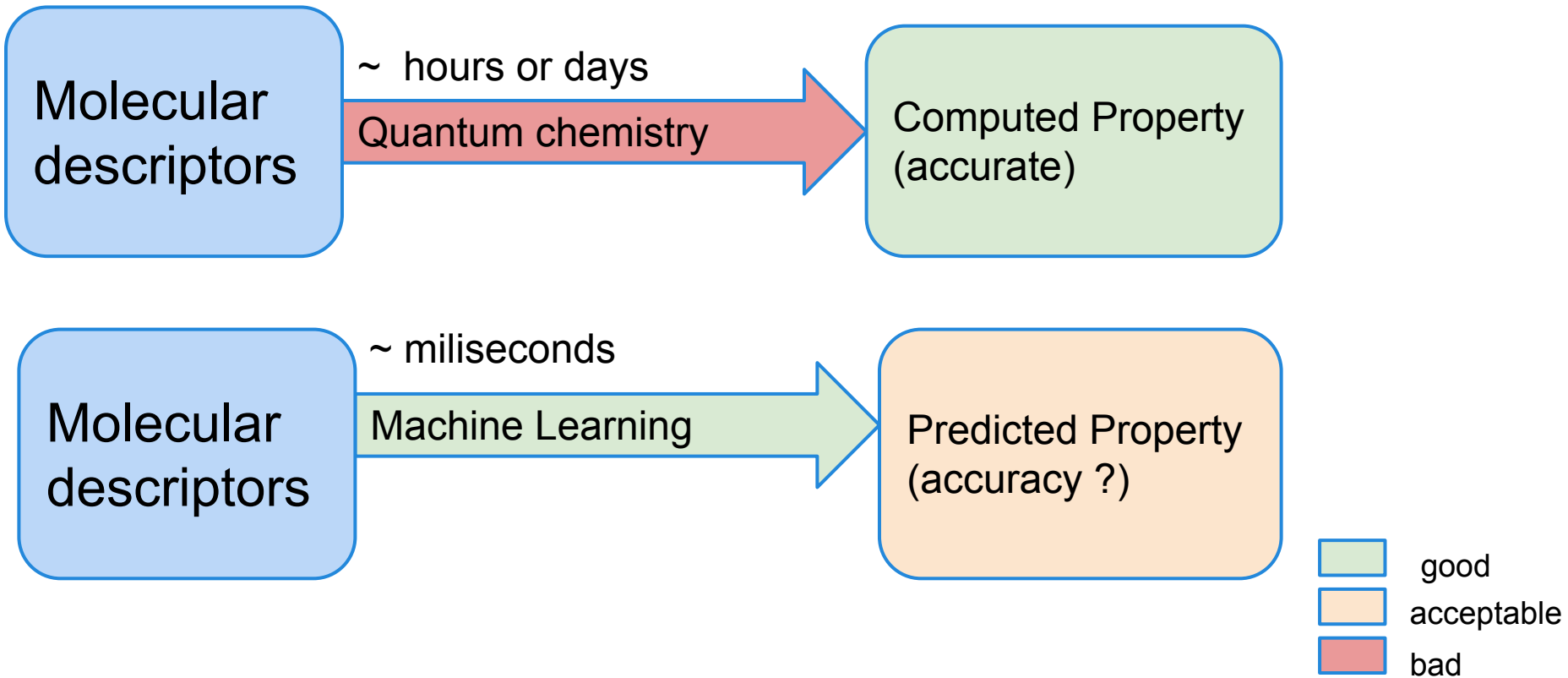
- Problem context
- Background Work
  - Learning Invariant Representations of Molecules for Atomization Energy Prediction
  - Self-taught Learning: Transfer Learning from Unlabeled Data
  - Information - Theoretic Regret Bounds for Gaussian Process Optimization in the Bandit Setting
- Conclusions

Overview

- **Problem context**
- Background Work
  - Learning Invariant Representations of Molecules for Atomization Energy Prediction
  - Self-taught Learning: Transfer Learning from Unlabeled Data
  - Information - Theoretic Regret Bounds for Gaussian Process Optimization in the Bandit Setting
- Conclusions

Problem context

- Compute/predict properties of molecules for materials design
  - Examples: drug discovery, water purification, energy transmission and storage
- Quantum Chemistry calculations are expensive
- Machine Learning could predict properties of molecules at a fraction of the cost


Molecular descriptors
~ hours or days
Quantum chemistry
Computed Property (accurate)
Molecular descriptors
~ miliseconds
Machine Learning
Predicted Property (accuracy ?)
good
acceptable
bad

Overview

- Problem context
- **Background Work**
  - **Learning Invariant Representations of Molecules for Atomization Energy Prediction**
  - Self-taught Learning: Transfer Learning from Unlabeled Data
  - Information - Theoretic Regret Bounds for Gaussian Process Optimization in the Bandit Setting
- Future work

Learning Invariant Representations of Molecules

- Desired properties of Molecular Descriptor
  - invariance to atom indexing
  - invariance to rotation and translation
- Coloumb Matrix [Rupp 2012] O(N^2)

$$C_{ij} = \begin{cases} 0.5 Z_i^{2.4} & \forall i = j \\ \frac{Z_i Z_j}{|\mathbf{R}_i - \mathbf{R}_j|} & \forall i \neq j. \end{cases}$$

Z_i - nuclear charges

$\mathbf{R_i}$ - 3D position

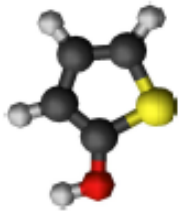
Coloumb Matrix

Coloumb Matrix descriptor

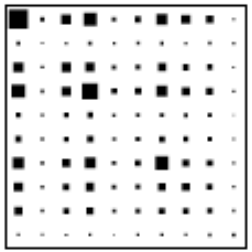
- invariant to rotation and translation

  (use $|\mathbf{R}_i - \mathbf{R}_j|$)
- invariance to atom indexing
  - Sorted Coloumb Matrix - indexes given by sorting the row norms
  - Random Coloumb Matrix - generate multiple Sorted Coloumb Matrices perturbed by noise)

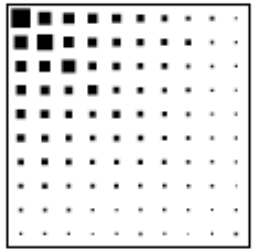
Random Coloumb Matrix



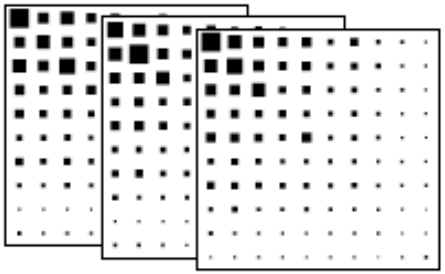
raw molecule



Coloumb Matrix



Sorted Coloumb Matrix



Randomly Sorted Coloumb Matrix

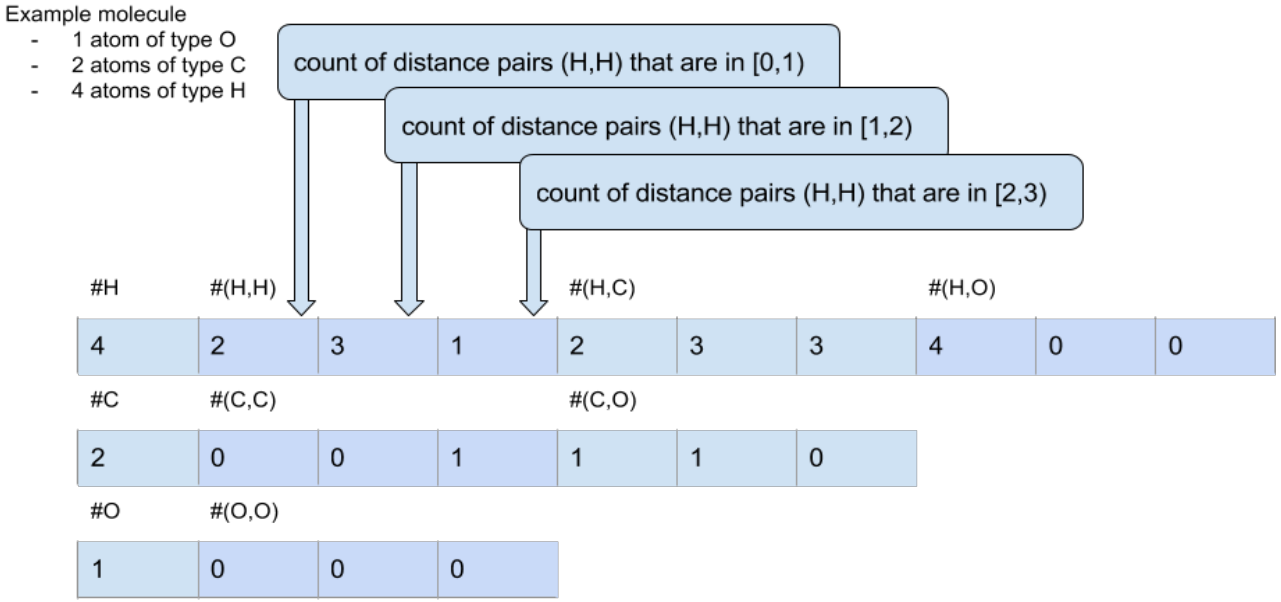
Atomization energy prediction for a dataset of ~7k samples (5.5 k training 1.5k testing) with {H,O,C,N,S}

- Multilayer Feed Forward NN with Random Coloumb

~3.1 kcal/mol MAE -> chemical accuracy level ~1kcal/mol

Proposed extension

- Obtain invariance to atom indexing through binning



Proposed extension

- Scales with with

$$NA + \sum_i \frac{N_A * (N_A + 1)}{2} * \frac{D_{max}}{q}$$

NA	number of atom types
q	quantization level
D_{max}	max distance in the dataset

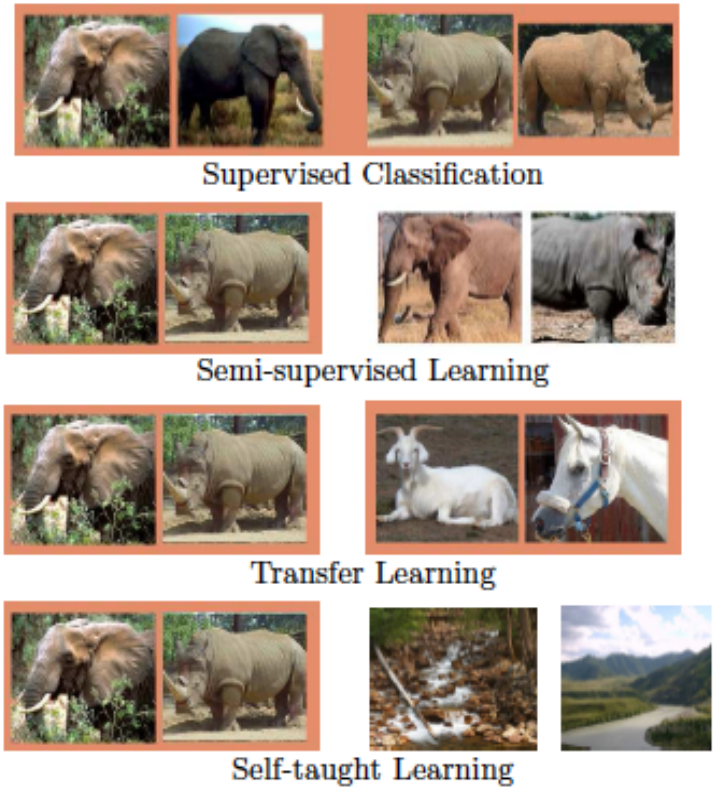
Overview

- Problem context
- **Background Work**
  - Learning Invariant Representations of Molecules for Atomization Energy Prediction
  - **Self-taught Learning: Transfer Learning from Unlabeled Data**
  - Information - Theoretic Regret Bounds for Gaussian Process Optimization in the Bandit Setting
- Conclusions

Self - Taught Learning: Transfer Learning from Unlabeled Data

- Supervised learning requires labeled data - labelling effort
- Large amounts of training data leads to better generalization performance
- ...but we can obtain more easily large amounts of unlabeled images, text documents…(molecular data?)

Self - Taught Learning

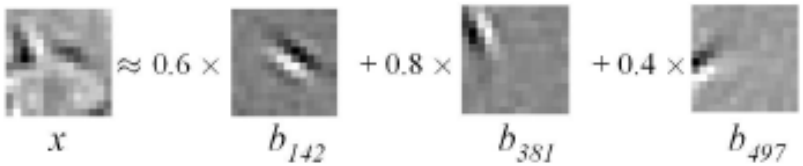


- Orange boxes contain labeled data
- Right side training data
- Left side testing data

Self - Taught Learning

Self - Taught learning algorithm

- Learning Higher-level Representations using sparse coding (find bases and sparse activations)



- The input feature is transformed in a feature vector containing the activations

Self - Taught Learning

Two optimization problems

- From unlabeled data, find bases b, activations a

$$\begin{aligned} \text{minimize}_{b,a} \quad & \sum_i \|x_u^{(i)} - \sum_j a_j^{(i)} b_j\|_2^2 + \beta \|a^{(i)}\|_1 \\ \text{s.t.} \quad & \|b_j\|_2 \leq 1, \quad \forall j \in 1, ..., s \end{aligned}$$

- For labeled data, using above bases, find sparse activations a

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

Experimental results

Extensive testing across a range of domains

- computer vision (raw pixels)
- natural language processing (bag of words)
- speech recognition (frequency histogram)

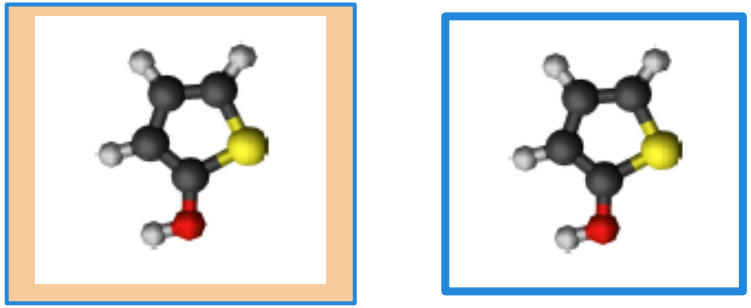
… molecular data ?

Proposed extension

- Avoid running expensive quantum chemistry calculations to get labeled data

- Labelling molecular data requires domain specific knowledge

Proposed extension



?

Example:

- Unlabeled dataset contains {H,C,O,N,S}
- Labeled test set contains {H,C,O,N,F}

Semi - supervised Learning

Transfer Learning

Self-Taught Learning

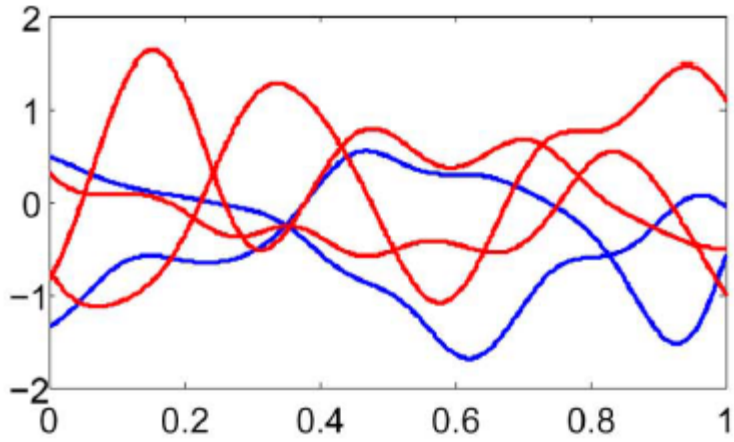
Overview

- Problem context
- **Background Work**
  - Learning Invariant Representations of Molecules for Atomization Energy Prediction
  - Self-taught Learning: Transfer Learning from Unlabeled Data
  - **Information - Theoretic Regret Bounds for Gaussian Process Optimization in the Bandit Setting**
- Conclusions

Gaussian Processes Introduction

Gaussian Processes

- define probabilities over functions f ~ GP(u, sigma)
- defined by the mean a



sample functions from squared exponential kernel

GP for function optimization

For T sampled points $A_T = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$ in the presence of noise

$$y_t = f(\boldsymbol{x}_t) + \epsilon_t$$

$$\boldsymbol{y}_T = [y_1 \cdots y_T]^T$$

$$\boldsymbol{x}^* = \text{argmax}_{\boldsymbol{x} \in D} f(\boldsymbol{x})$$

$$r_t = f(\boldsymbol{x}^*) - f(\boldsymbol{x}_t) \qquad R_T = \sum_{t=1}^{T} r_t$$

no-regret: $\lim_{T \to \infty} R_T / T = 0$

GP for function optimization (introduce regret)

Posterior is also Gaussian

$$y_t = f(\boldsymbol{x}_t) + \epsilon_t \qquad \boldsymbol{y}_T = [y_1 \cdots y_T]^T$$
$$A_T = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$$

$$\mu_T(\boldsymbol{x}) = \boldsymbol{k}_T(\boldsymbol{x})^T (\boldsymbol{K}_T + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{y}_T$$
$$k_T(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x}, \boldsymbol{x}') - \boldsymbol{k}_T(\boldsymbol{x})^T (\boldsymbol{K}_T + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{k}_T(\boldsymbol{x}')$$
$$\sigma_T^2(\boldsymbol{x}) = k_T(\boldsymbol{x}, \boldsymbol{x})$$

Function optimization

$$\boldsymbol{x}^* = \operatorname{argmax}_{\boldsymbol{x} \in D} f(\boldsymbol{x})$$

$$r_t = f(\boldsymbol{x}^*) - f(\boldsymbol{x}_t) \qquad R_T = \sum_{t=1}^{T} r_t$$

no-regret: $\lim_{T \to \infty} R_T / T = 0$

GP-UCB algorithm

Algorithm 1 The GP-UCB algorithm.

Input: Input space D; GP Prior $\mu_0 = 0, \sigma_0, k$

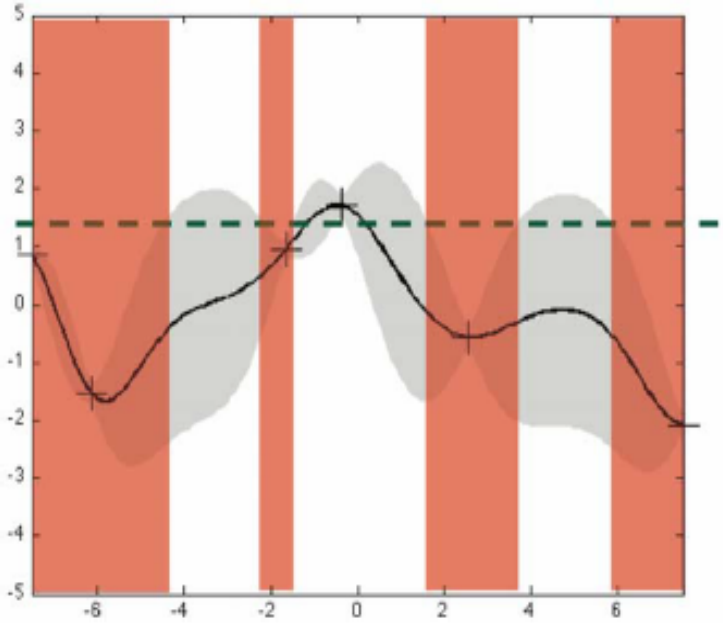
for $t = 1, 2, \ldots$ **do**

Choose $\boldsymbol{x}_t = \underset{\boldsymbol{x} \in D}{\operatorname{argmax}}\, \mu_{t-1}(\boldsymbol{x}) + \sqrt{\beta_t}\sigma_{t-1}(\boldsymbol{x})$

Sample $y_t = f(\boldsymbol{x}_t) + \epsilon_t$

Perform Bayesian update to obtain μ_t and σ_t

end for



Information Gain

Information Gain (Learn f as fast as possible) $y_A = f_A + \epsilon_A \qquad A \subset D, \ |A| \leq T$

$$\mathrm{I}(\boldsymbol{y}_A; f) = \mathrm{H}(\boldsymbol{y}_A) - \mathrm{H}(\boldsymbol{y}_A | f)$$

Near optimal solution approximation for max $F(A) = \mathrm{I}(\boldsymbol{y}_A; f)$

$$\boldsymbol{x}_t = \mathrm{argmax}_{\boldsymbol{x} \in D} F(A_{t-1} \cup \{\boldsymbol{x}\}) \text{ in round } t$$

$$\boldsymbol{x}_t = \underset{\boldsymbol{x} \in D}{\mathrm{argmax}}\, \sigma_{t-1}(\boldsymbol{x}) \qquad A_{t-1} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{t-1}\}$$

Regret bounds

Bound regret using

$$\gamma_T := \max_{A \subset D : |A| = T} \mathrm{I}(\boldsymbol{y}_A; \boldsymbol{f}_A)$$

Bound max IG using eigenspectrum of KD

Derive anayltical bounds on eigenspectrum of KD for most popular kernel types

Regret bound

Kernel	Linear	RBF	Matérn
Regret R_T	$d\sqrt{T}$	$\sqrt{T(\log T)^{d+1}}$	$T^{\frac{\nu+d(d+1)}{2\nu+d(d+1)}}$

Fig. 1. Our regret bounds (up to polylog factors) for linear, radial basis, and Matérn kernels—d is the dimension, T is the time horizon, and ν is the Matérn parameter.

Proposed extension

Either use them to propose new sampling [1] rather not scalable

Mostly used in hyper parameter optimization -

x represents the model parameters and f the cross validation value.

TODO introdue GP runs in the report

Thank you!