Xiamen University

cecam

Centre Européen de Calcul Atomique et Moléculaire

General introduction to machine learning

Pavlo O. Dral

dral@xmu.edu.cn; dr-dral.com

Xiamen University, China

MLatom.com

XACS

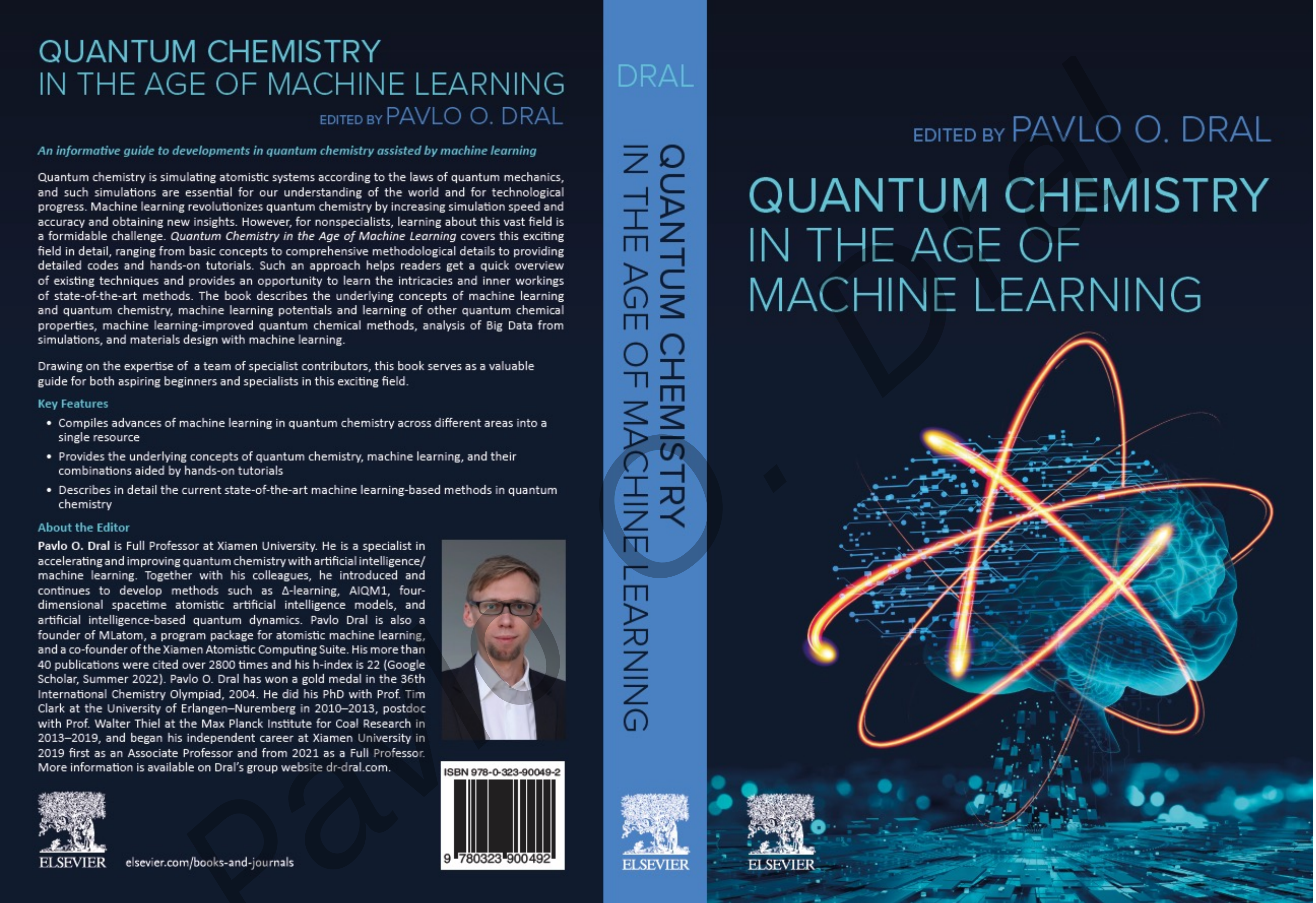
MLQCDyn

reinventing dynamics

A Package for
Atomistic Simulations
with **M**achine **L**earning

6 September 2022

Please ask questions during the lecture



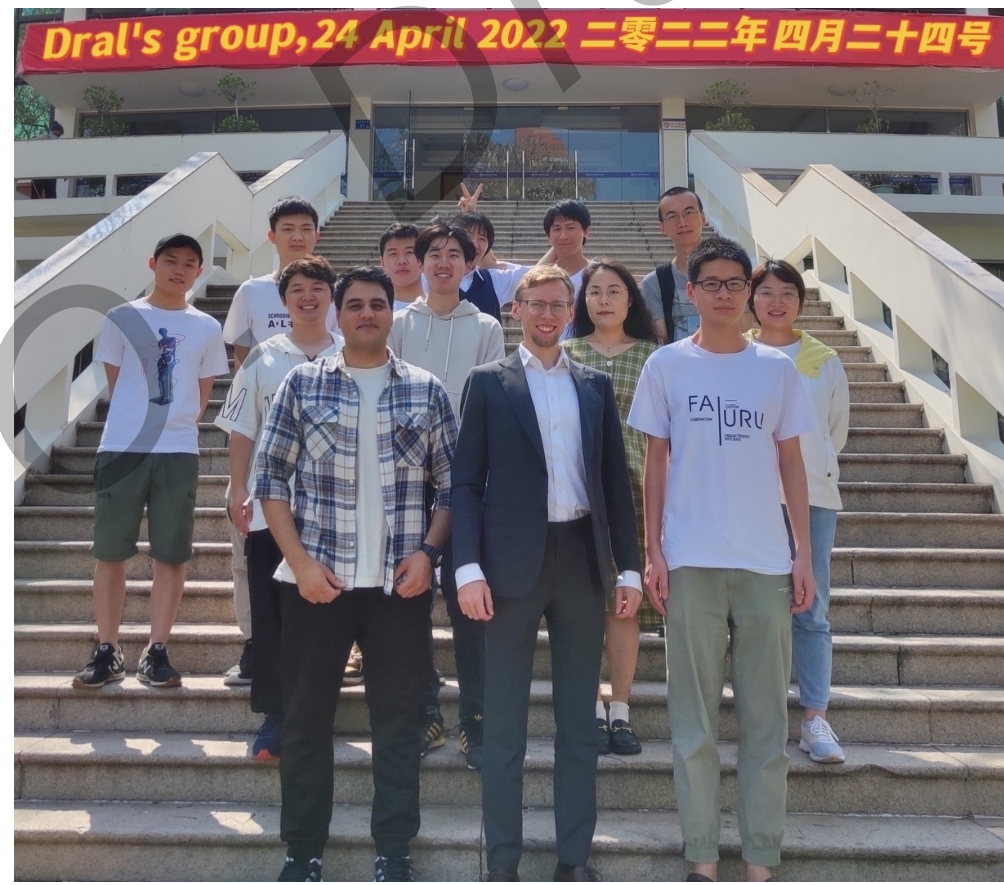
Book to be published on 16 September 2022

CV & Our group

2021: Outstanding Youth (Overseas), National Natural Science Foundation of China

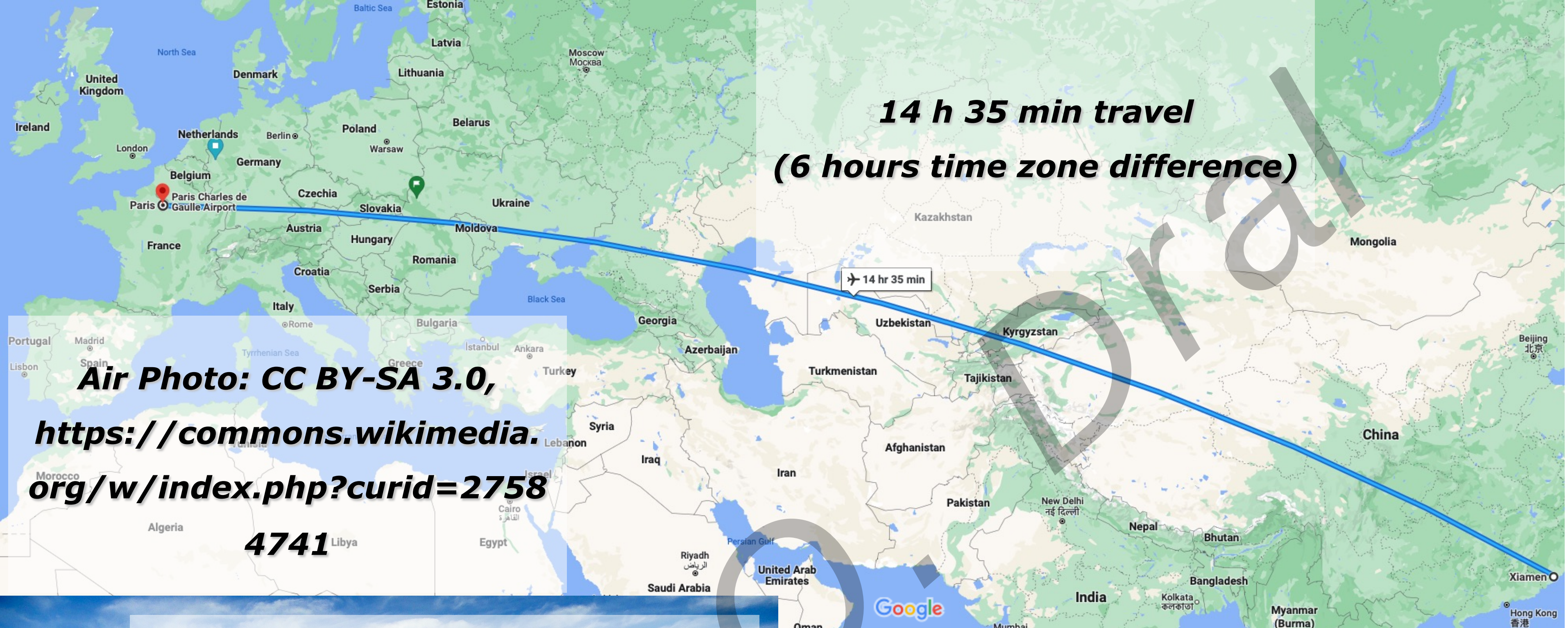
Hiring post-docs, PhD & MSc!

From 12.2021	Full Professor, Xiamen University
2019–2021	Associate Professor, Xiamen University
2013–2019	Post-doc, Max-Planck-Institut für Kohlenforschung (Walter Thiel)
2010–2013	Dr. rer. nat. in chemistry, University of Erlangen-Nürnberg (Timothy Clark)
2008–2010	Mag. in chemical technology and engineering, National Technical University of Ukraine "Kiev Polytechnic Institute" (Andrey A. Fokin)
2008–2010	M. Sc. in molecular science, University of Erlangen-Nürnberg (Timothy Clark)



14 h 35 min travel

(6 hours time zone difference)



Air Photo: CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=27584741

Ca. 2 mln people on Xiamen Island

16 subjects rank top 1% globally. 11th in Mainland China.

Chemistry ranks top 1‰ globally (*ESI as of March 2019*).

- Chemistry
- Engineering
- Material Science
- Clinical Medicine
- Plant and Zoology
- Environment and Ecology
- Biology and Biochemistry
- Molecular Biology & Genetics
- Pharmacology &Toxicology
- Computer Science
- Earth Science
- Clinical Medicine
- Microbiology
- Mathematics
- Agriculture
- Physics

ML in Chemistry

What year was this Review published?

Review

Neural networks: A new method for solving chemical problems or just a passing phase?

Authors??

Recent work on **neural networks in chemistry** is reviewed and essential background to this fast-spreading method is given. Emphasis is placed on the back-propagation algorithm, because of the extensive use of this form of learning. Hopfield networks, adaptive bidirectional associative memory, and Kohonen learning are briefly described and discussed. **Applications in spectroscopy (mass, infrared, ultraviolet, NMR), potentiometry, structure/activity relationships, protein structure, process control and chemical reactivity are summarized.**

ML in Chemistry

Analytica Chimica Acta, 248 (1991) 1–30
Elsevier Science Publishers B V , Amsterdam

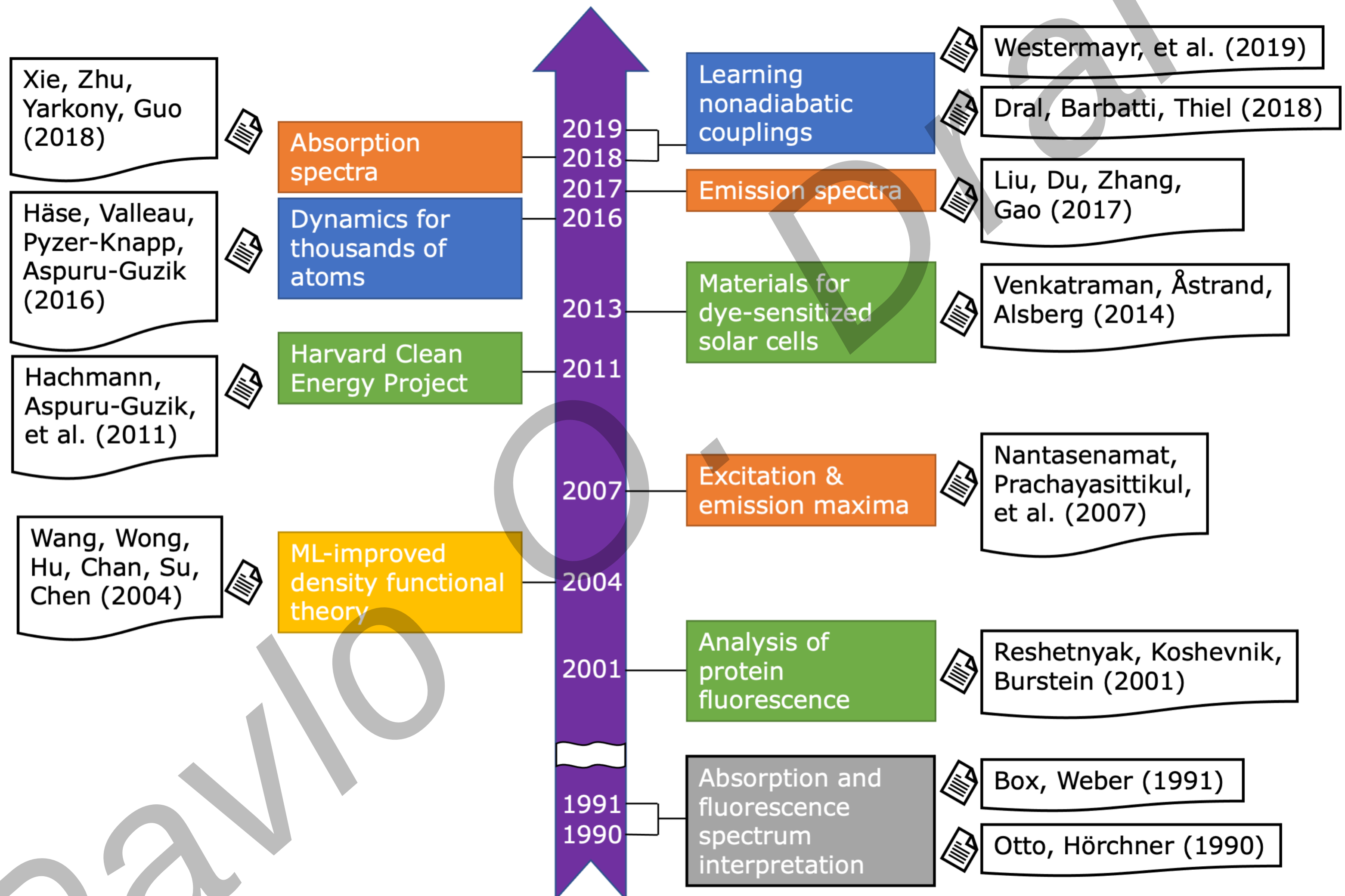
Review

Neural networks: A new method for solving chemical problems or just a passing phase?

J. Zupan [*,1] and J. Gasteiger

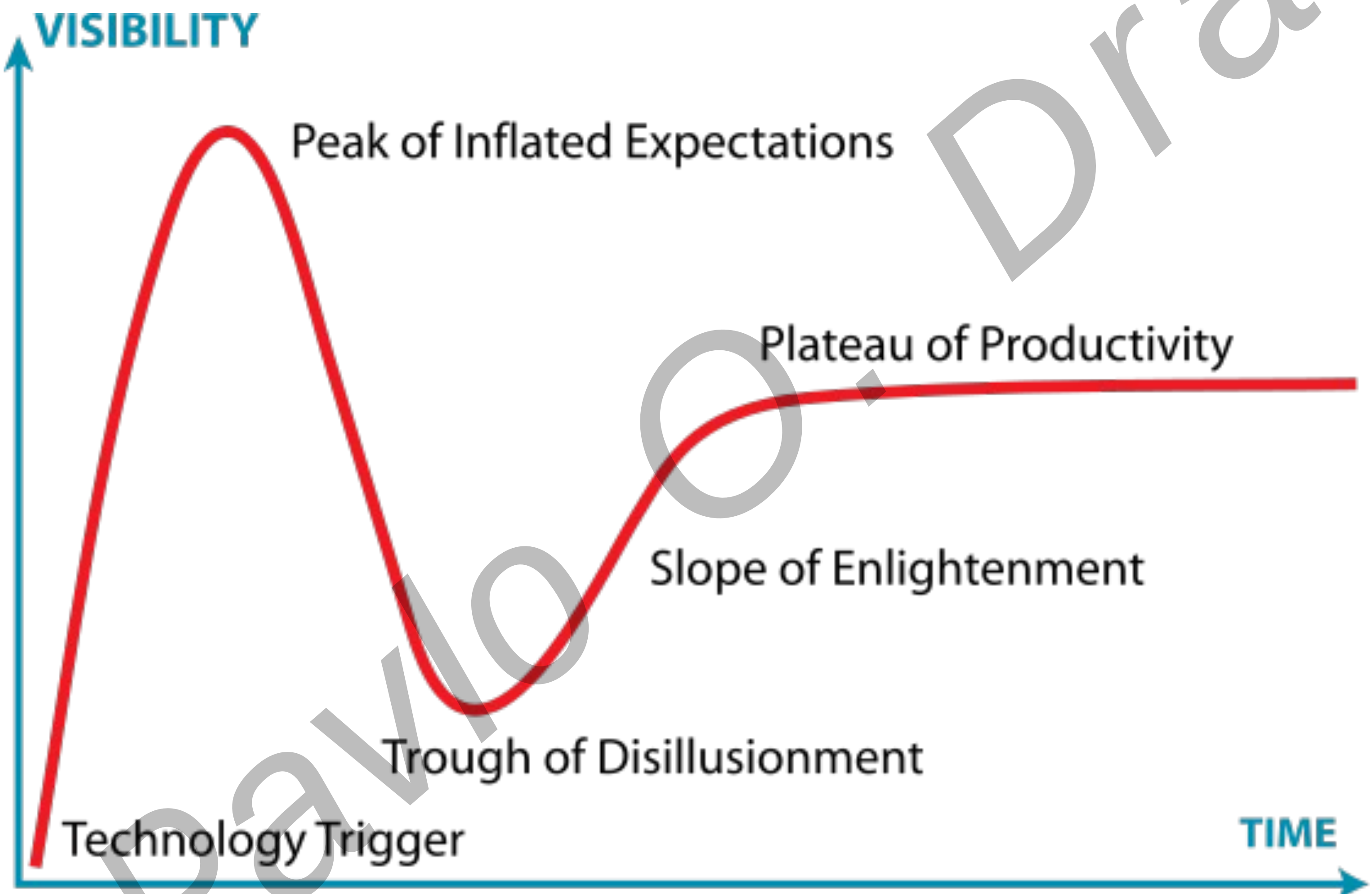
Recent work on **neural networks in chemistry** is reviewed and essential background to this fast-spreading method is given. Emphasis is placed on the back-propagation algorithm, because of the extensive use of this form of learning. Hopfield networks, adaptive bidirectional associative memory, and Kohonen learning are briefly described and discussed. **Applications in spectroscopy (mass, infrared, ultraviolet, NMR), potentiometry, structure/activity relationships, protein structure, process control and chemical reactivity are summarized.**

ML for excited states



P. O. Dral, M. Barbatti, *Nat. Rev. Chem.* **2021,** *5*, 388

Gartner's Hype Cycle



Jeremykemp at English Wikipedia [GFDL or CC BY-SA], from Wikimedia Commons

Example: ML in Chess

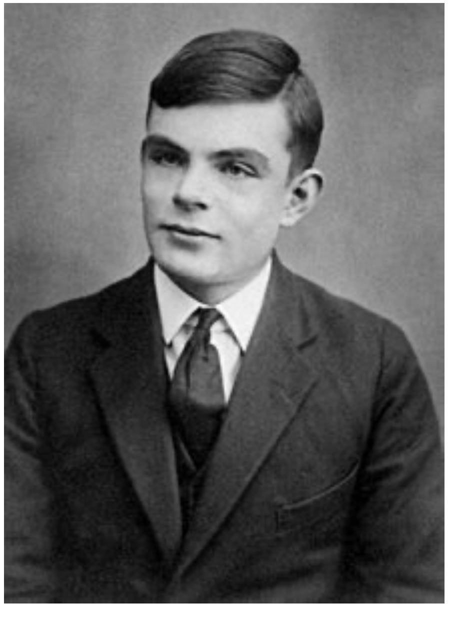
Number of possible move sequences in chess is very big:

No brute-force solution of chess is possible

Example: ML in Chess

- Making machines play chess was considered by the pioneers of AI as an **important milestone for AI**.

- In **1950s** the first computer chess programs were developed with contributions by Alan Turing.



- In **1997** computer has for the first time defeated the best human player in a match

Example: ML in Chess

But are the computer chess programs **intelligent**?

The world champion Magnus Carlsen in 2017: "The problem is that it still feels like you are playing somebody **stupid** when you are playing the computer."

Photo by Andreas Kontokanis from Piraeus, Greece (Carlsen Magnus) [CC BY-SA 2.0], via Wikimedia Commons

Example: ML in Chess

The chess programs Magnus Carlsen has been talking about are **coded explicitly by humans**. Algorithms and hardware have been improved ca. **70 years**.

Can ML beat 70 years of human effort?

Machine learning

The name ‘Machine learning’ (ML) implies that machines try to learn from [Big] data by themselves **without being programmed explicitly** by humans.

ML is an application of a broader artificial intelligence (AI) field to practical problems.

Example: ML in Chess

On 5 December 2017 a pre-print by Google's DeepMind team[1] has been published about AlphaZero ML-based program:[2]

- It was trained only by self-play for **8 hours**
- It could beat the best computer chess program in a match
- Chess grandmasters describe AlphaZero's play as **human-like** and were impressed by its positional and attacking play

[1] D. Hassabis et al., *arXiv:1712.01815*, **2017**
[2] D. Hassabis et al., *Science* **2018**, *362*, 1140

Power of ML and What It Needs

This is a power of ML: it can (self-)learn so fast that it delivers better results than programs coded explicitly by humans over decades!

Power of ML and What It Needs

This is a power of ML: it can (self-)learn so fast that it delivers better results than programs coded explicitly by humans over decades!

The “only” thing it requires is lots of data and good hardware: we have it all nowadays

Chemistry and Big Data

Chemistry is rich in Big Data:

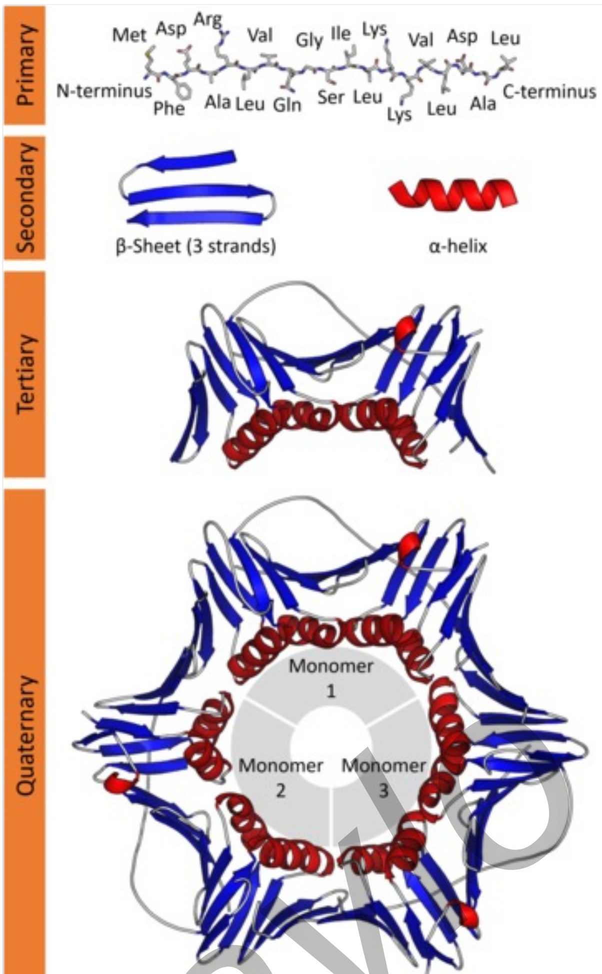
- Number of possible compounds is infinite
- Number of points on a potential energy surface (PES) is infinite

ML in Chemistry

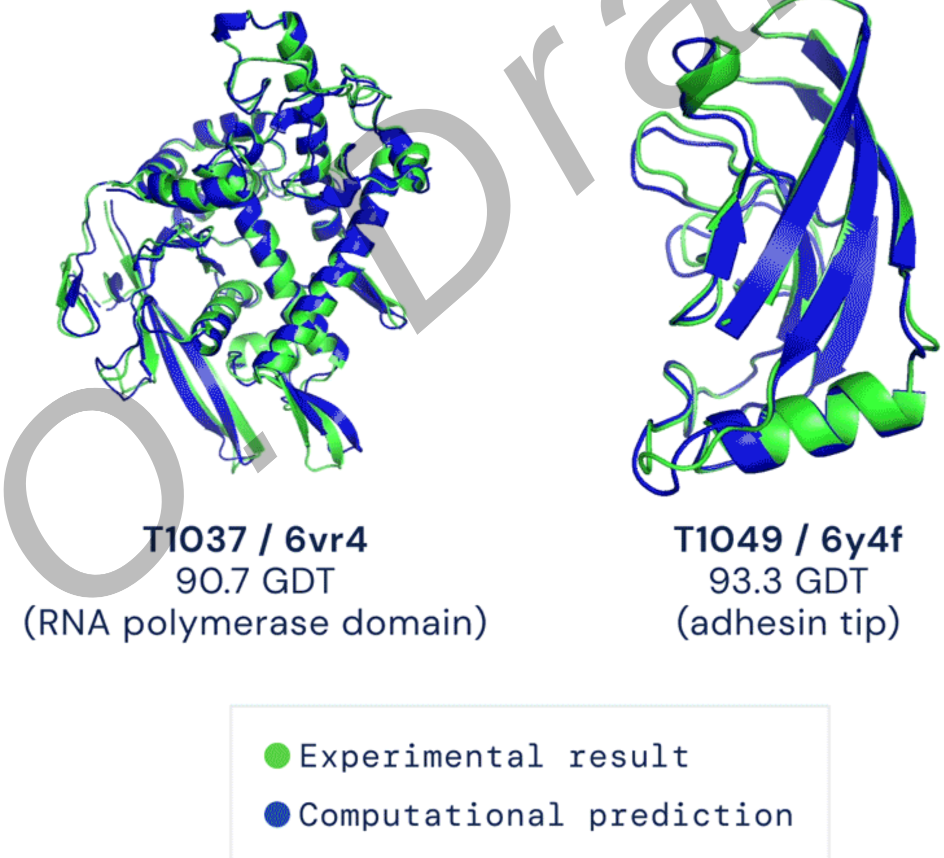
Chemistry is more difficult than chess:

After many years of ML in chemistry we are only starting to use its potential

ML in Chemistry: AlphaFold (2)



Thomas Shafee, CC BY 4.0, via Wikimedia Commons



Video from https://deepmind.com/blog/article/alphafold-a-solution-to-a-50-year-old-grand-challenge-in-biology

DeepMind

Quantum Chemistry

Time-independent Schrödinger equation

$$\hat{H}\Psi = E\Psi$$

$$\hat{H} = -\sum_{i=1}^{N}\frac{1}{2}\nabla_i^2 - \sum_{A=1}^{M}\frac{1}{2M_A}\nabla_A^2 - \sum_{i=1}^{N}\sum_{A=1}^{M}\frac{Z_A}{r_{i,A}} + \sum_{i=1}^{N}\sum_{j>i}^{N}\frac{1}{r_{i,j}} + \sum_{A=1}^{M}\sum_{B>1}^{M}\frac{Z_A Z_B}{R_{A,B}}$$

$\hat{H}$ - Hamiltonian
Ψ – wavefunction
E - energy

Bond Length in H_2

Experiment: 0.7414 Å

Quantum Chemistry: 0.7415 Å, ~5 CPU-days
(FCI/aug-cc-pV6Z)

Quantum Chemistry

The Schrödinger equation

$$\hat{H}\Psi = E\Psi$$

$$\hat{H} = -\sum_{i=1}^{N}\frac{1}{2}\nabla_i^2 - \sum_{A=1}^{M}\frac{1}{2M_A}\nabla_A^2 - \sum_{i=1}^{N}\sum_{A=1}^{M}\frac{Z_A}{r_{i,A}} + \sum_{i=1}^{N}\sum_{j>i}^{N}\frac{1}{r_{i,j}} + \sum_{A=1}^{M}\sum_{B>1}^{M}\frac{Z_A Z_B}{R_{A,B}}$$

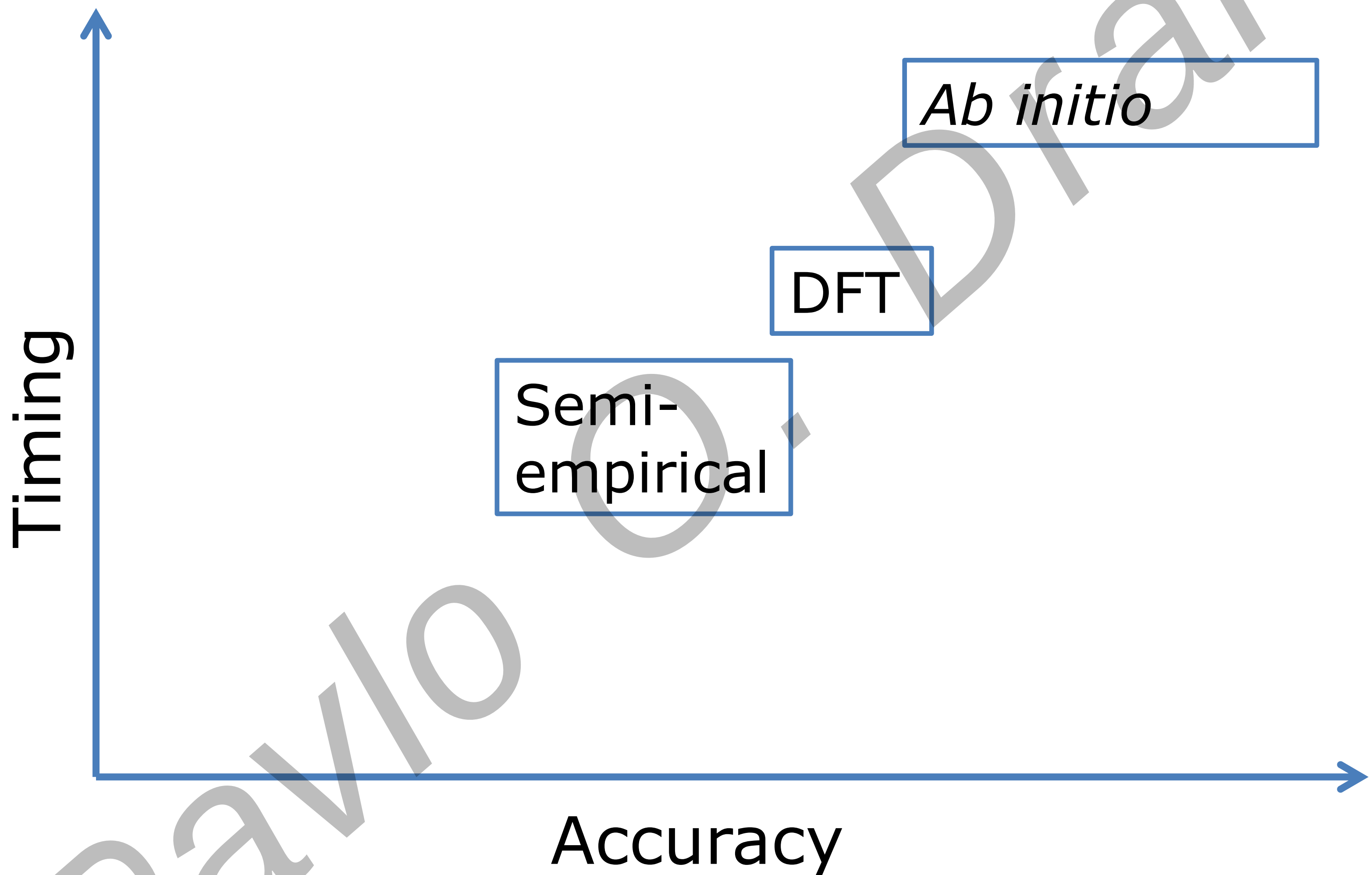
$\Downarrow$

The electronic Schrödinger equation can be solved exactly ***only*** for two-body systems (e.g. hydrogen atom with 1 nucleus + 1 electron)

$\Downarrow$

Approximations are needed!

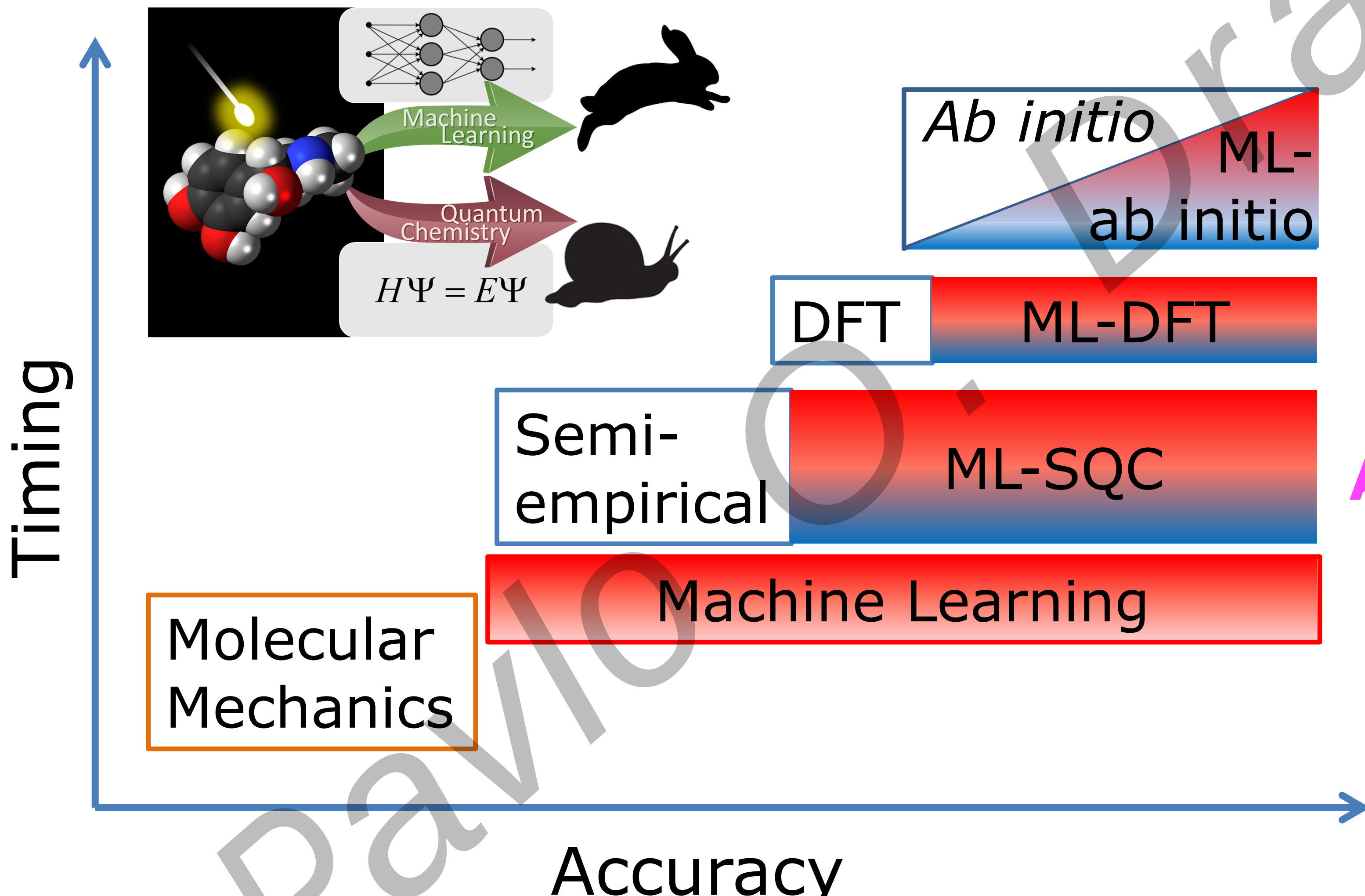
Quantum chemical methods



P. O. Dral, *J. Phys. Chem. Lett.* **2020**, *11*, 2336

Computational Chemistry Methods

P. O. Dral, M. Barbatti, *Nat. Rev. Chem.* **2021,** *5*, 388



PS3

AIQM1

P. Zheng, R. Zubatyuk, W. Wu, O. Isayev, P. O. Dral, *Nat. Commun.* **2021**, *12*, 7022

P. O. Dral, *J. Phys. Chem. Lett.* **2020**, *11*, 2336

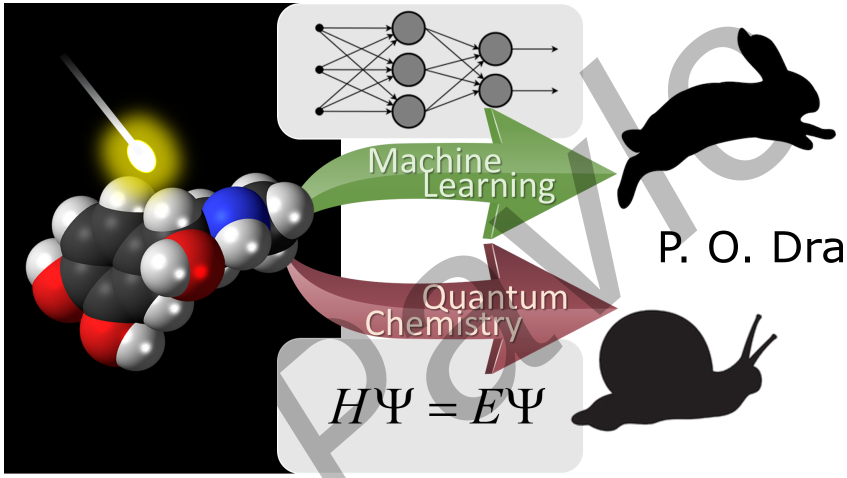
Bond Length in H_2

Experiment: 0.7414 Å

Quantum Chemistry: 0.7415 Å, ~5 CPU-days
(FCI/aug-cc-pV6Z)

Machine learning (ML): 0.7415 Å, ~0.3 seconds

PS3



P. O. Dral, M. Barbatti, *Nat. Rev. Chem.* **2021**, *5*, 388

Quantum chemistry vs Machine learning

Conventional programming in quantum chemistry:

→ Code for molecular orbitals

→ Code for excitation energies

→ Code for oscillator strengths

→ …

Machine learning (in principle – adaptations required!):

→ The same code for all above

MOs ↑ excitation energies ↑ Oscillator strength ↑

Training data

Machine learning

The name 'Machine learning' (ML) implies that machines try to learn from [Big] data by themselves **without being programmed explicitly** by humans.

ML is an application of a broader artificial intelligence (AI) field to practical problems.

Quantum chemistry vs Machine learning

Quantum Chemistry

a) solve $\hat{H}\Psi = E\Psi$ → Training Data

b) calculate E_{ML} *etc.*

c) solve $\hat{H}_{ML}\Psi = E\Psi$

d) predict and use Ψ_{ML}

Machine Learning

Machine learning potentials (MLPs) – L5

P. O. Dral, *J. Phys. Chem. Lett.* **2020**, *11*, 2336

Types of Machine Learning

Types of Machine Learning

Supervised Machine Learning

Supervised Machine Learning

Input (x) → f(x) → Output (y)

Given collection of known {x,y} find a function f(x)

Supervised Machine Learning

Input (x) → *f*(x) → Output (y)

↓

Given collection of known {x,y} find a function *f*(x)

↓

Use this function for making new predictions given just {x’}

Supervised Machine Learning

Input (x) → *f*(x) → Output (y)

Given collection of known {x,y} find a function *f*(x)

training set   train   ML model

Use this function for making new predictions given just {x'}

Unsupervised Machine Learning

Types of Machine Learning

Unsupervised Machine Learning

Only input (x)

Types of Machine Learning

Unsupervised Machine Learning

Only input (x)

Find patterns

Find correlations

Divide *x* into clusters

Types of Machine Learning

Supervised Machine Learning

Input (x) → *f*(x) → Output (y)

Given collection of known {x,y} find a function *f*(x)

training set train ML model

Use this function for making new predictions given just {x'}

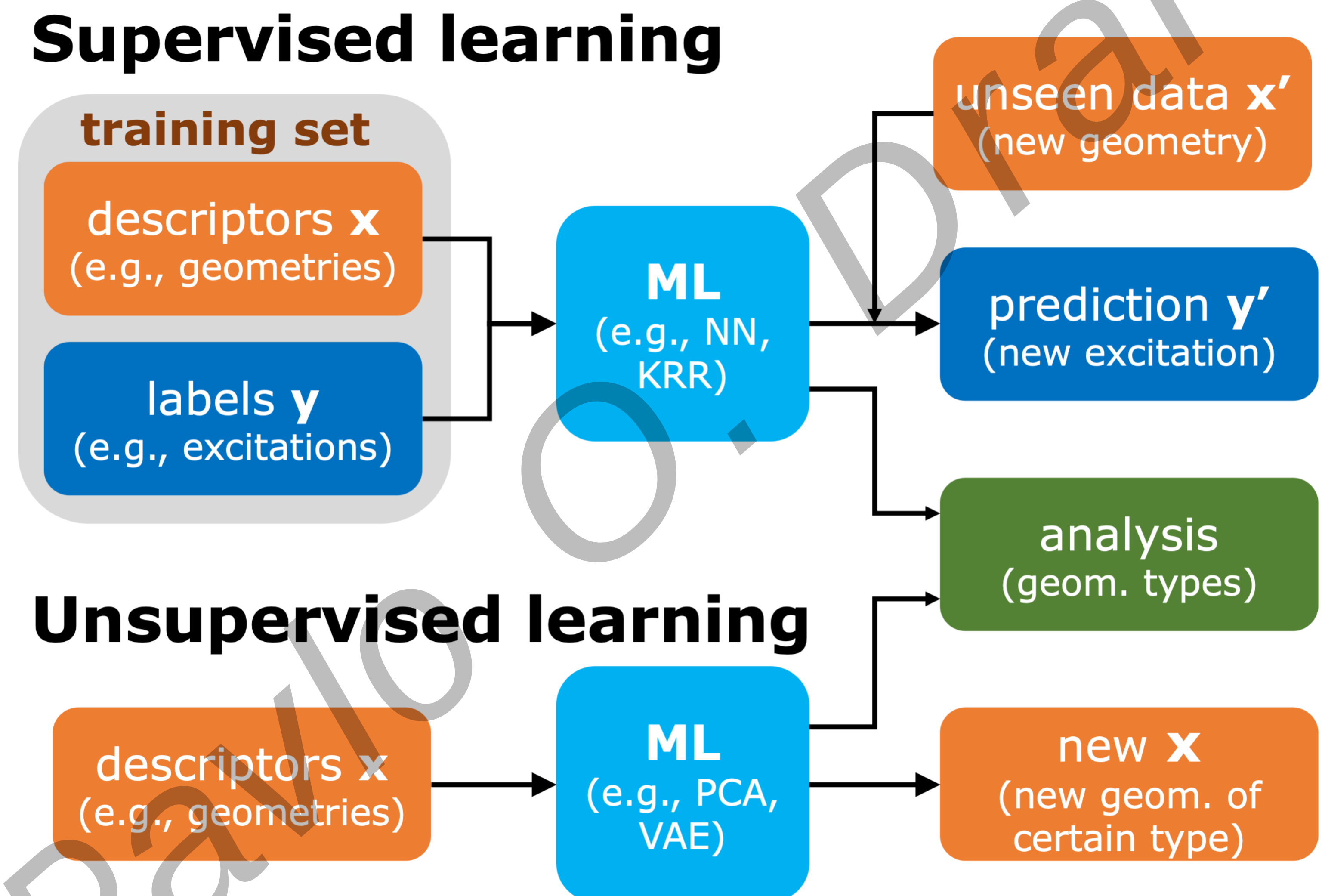
Unsupervised Machine Learning

Only input (x)

Find correlations

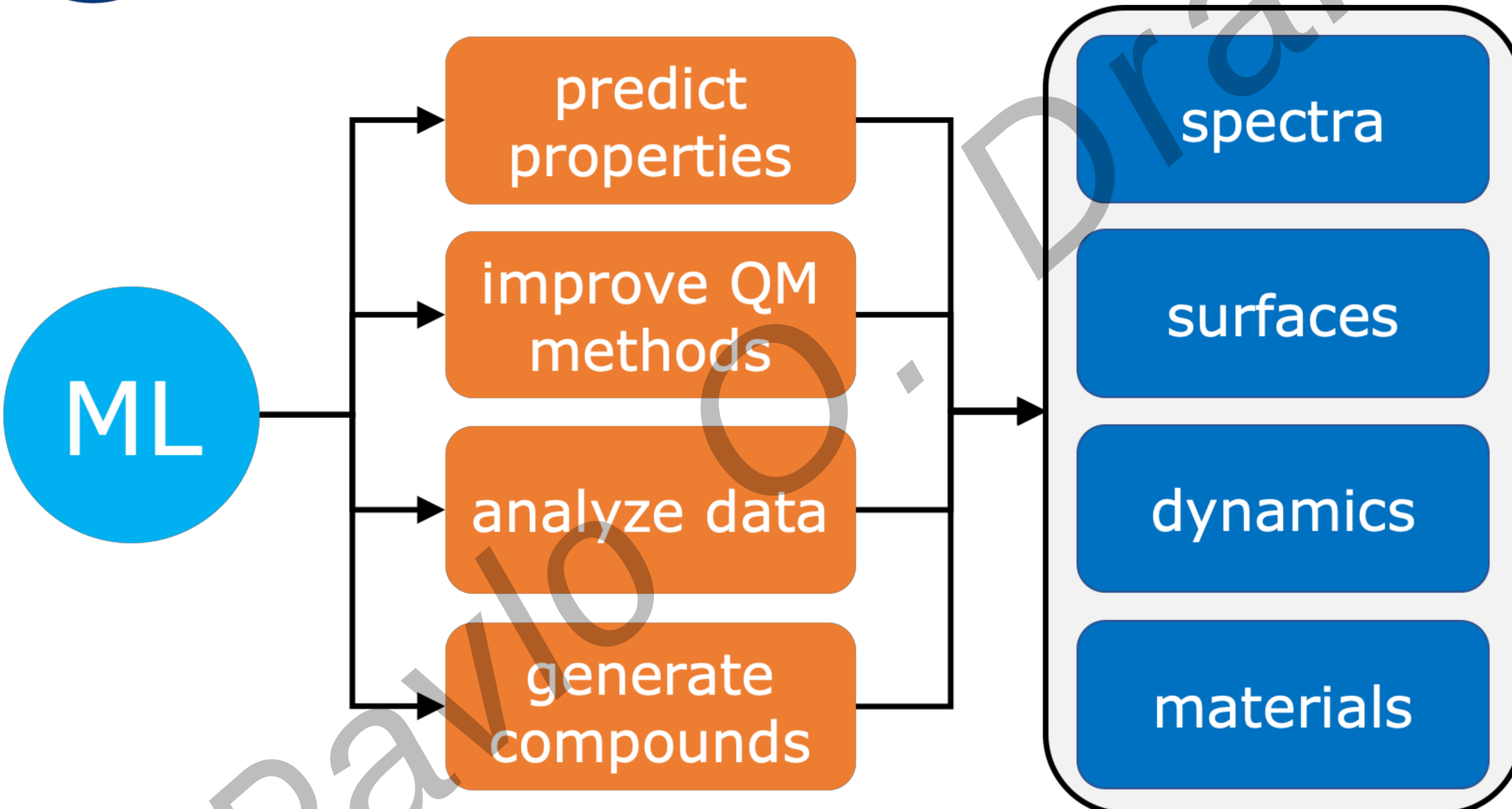
Divide *x* into clusters

ML in Chemistry



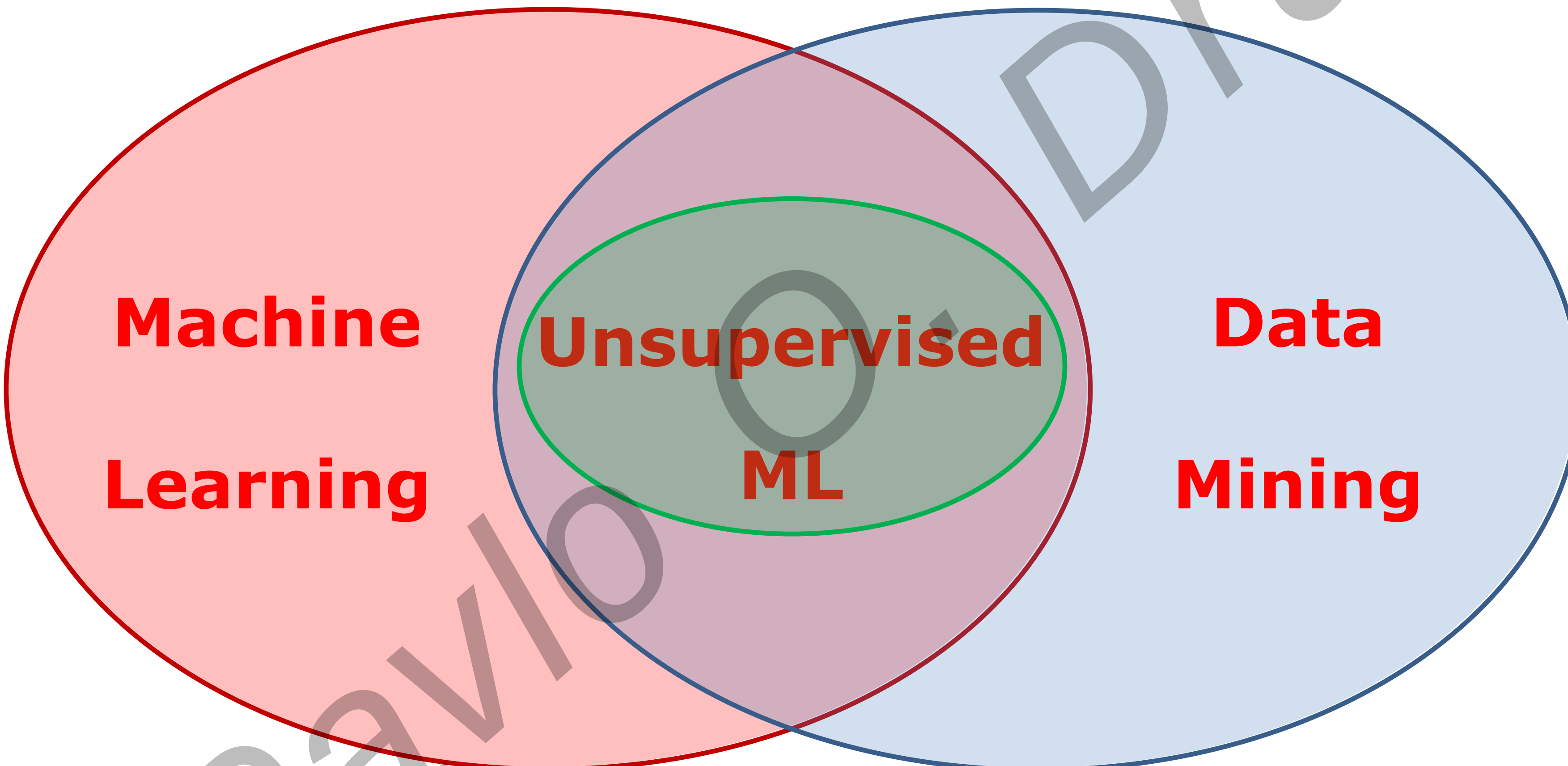
P. O. Dral, M. Barbatti, *Nat. Rev. Chem.* **2021,** *5*, 388

Machine learning for excited states



P. O. Dral, M. Barbatti, *Nat. Rev. Chem.* **2021,** *5*, 388

Types of Machine Learning

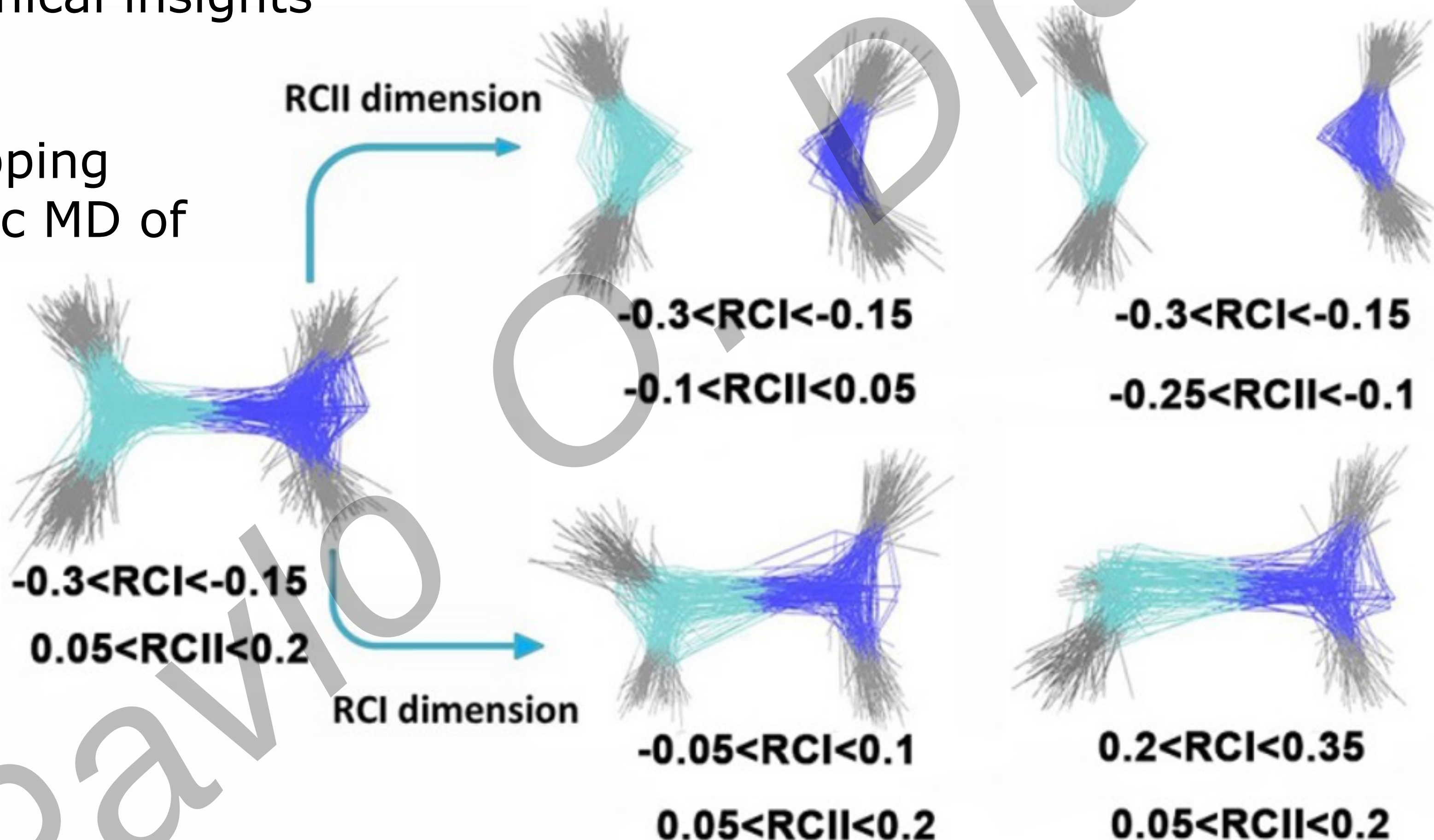


Unsupervised ML for MD

Unsupervised ML is useful for analyzing MD trajectories and getting physicochemical insights

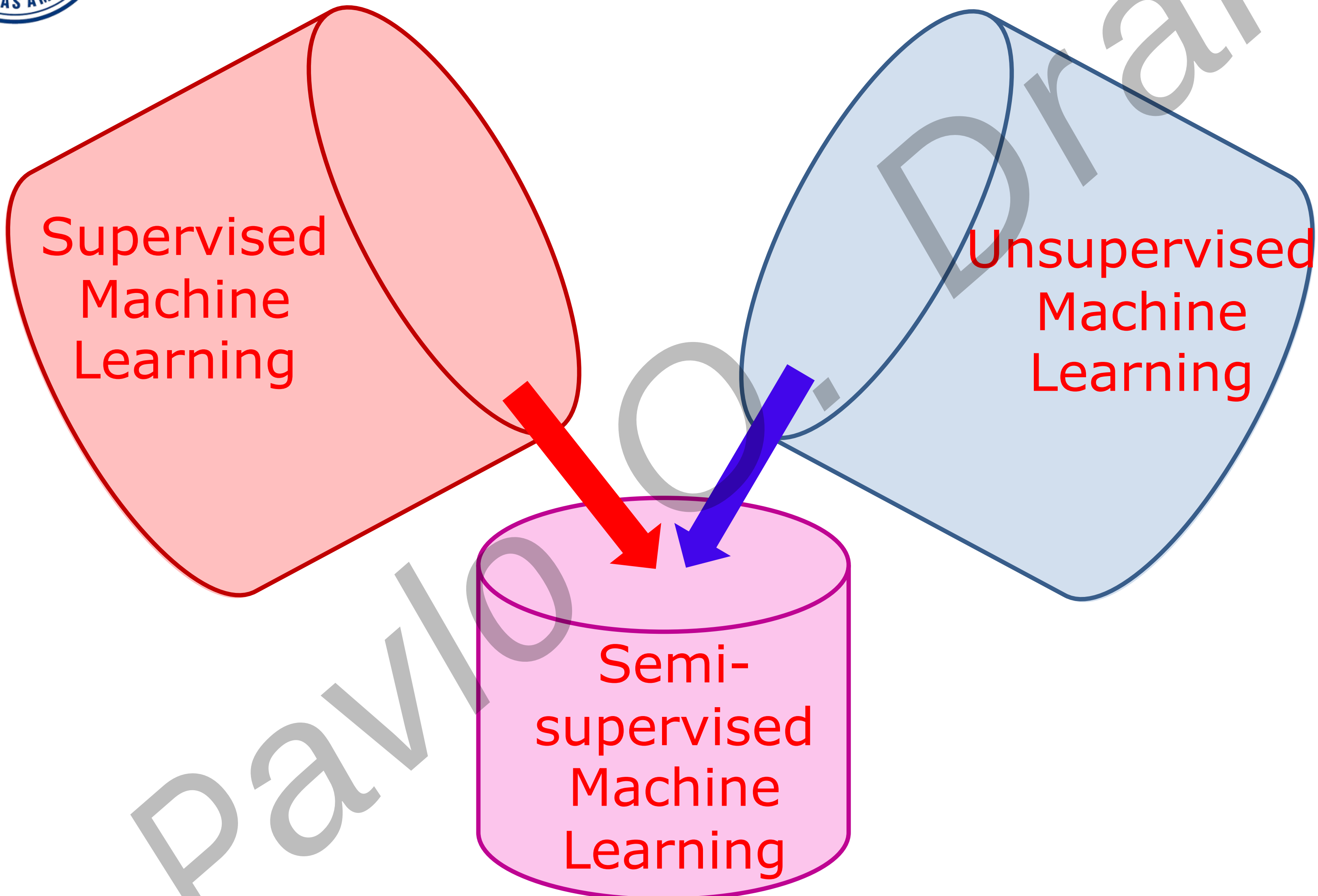
On-the-fly surface-hopping nonadiabatic MD of $CH_2NH_2^+$

PS6



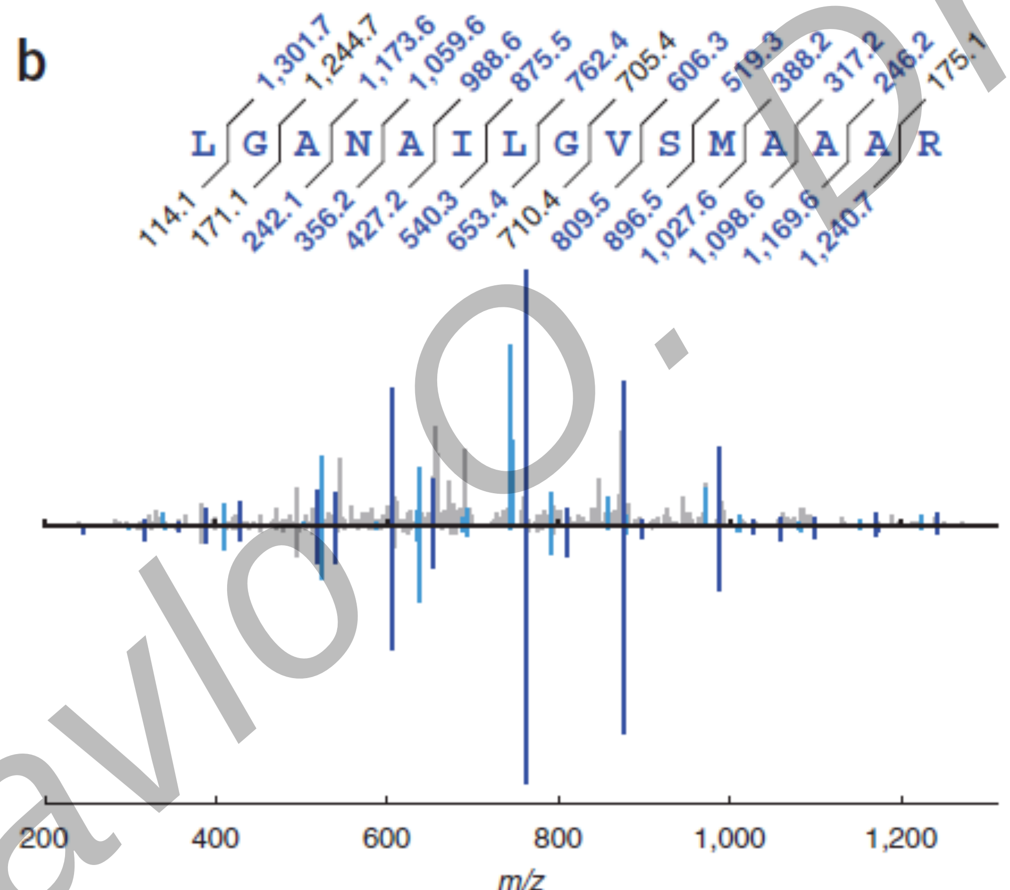
Li, Xie, Hu, Lan, *J. Chem. Theory Comput.* **2017**, *13*, 4611

Types of Machine Learning



Semi-supervised ML in Chemistry

Semi-supervised ML is often used in protein research,
e.g. it can improve identification of correct peptide from mass-spectra



Kall, Canterbury, Weston, Noble, MacCoss, *Nature Methods* **2007**, *4*, 923

Types of Machine Learning

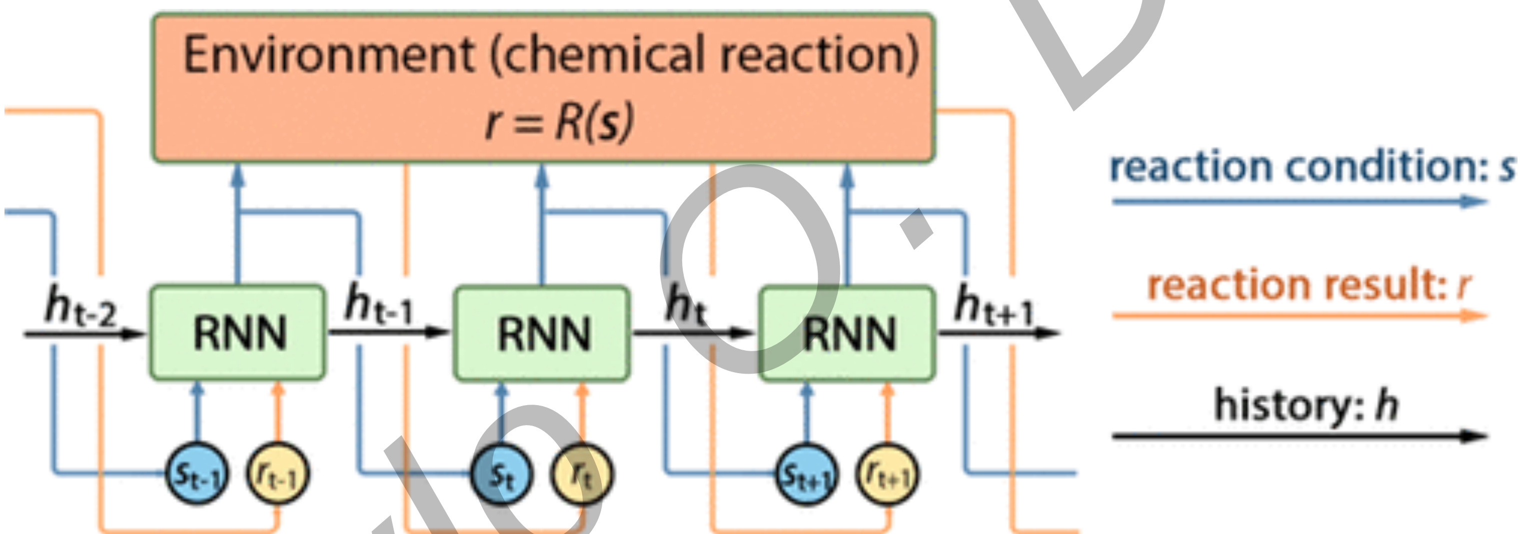
Reinforcement Learning

ML tries to maximize rewards from environment

In case of chess:
Tries to maximize the number of wins

Reinforcement Learning

For example, reinforcement learning can be used in chemistry for optimizing chemical reactions



Zhou, Li, Zare, *ACS Cent. Sci.* **2017**, *3*, 1337

Types of Machine Learning

Supervised Machine Learning

Input (x) → *f*(x) → Output (y)

Given collection of known {x,y} find a function *f*(x)

training set train ML model

Use this function for making new predictions given just {x'}

- Data
- Choice of x (descriptor)
- Choice of y (labels)
- Fitting function (ML algorithm, ML model)
- Optimization of ML model parameters

Chemistry and Machine Learning

(Supervised)
Machine learning serves for function approximation[1]

ML takes little time for making new predictions

Quantum Chemical Property(molecule) = function(nuclear coordinates)

[1] Hastie, Tibshirani, Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed.; Springer-Verlag, **2009**

What we need to have:

- Data
- QC Model (Hamiltonian)
  vs
  ML Model (not constrained by physical model)
- Parameters

What we need to do:

- Fit parameters to achieve an optimization goal

The difference between QC and ML models

- The goal of QC model (Hamiltonian) is to be physically correct as much as possible
- The goal of ML Model is to generalize (not just to fit!) from data as good as possible

Q: What parameters are in HF/3-21G?

Parameters in Hartree–Fock/3-21G

Parameters (in red) are in the basis set, e.g., for hydrogen at 3-21G:

$$\varphi'_{1s}(\mathbf{r}) = \sum_{i=1}^{2} d'_{i,1s} \left(\frac{8\alpha'^3}{\pi^3}\right)^{1/4} \exp(-\alpha' r^2)$$

$$\varphi''_{1s}(\mathbf{r}) = \sum_{i=1}^{2} \left(\frac{8\alpha''^3}{\pi^3}\right)^{1/4} \exp(-\alpha'' r^2)$$

How this parameters were obtained?
By optimizing them so that SCF atomic energy reaches minimum

Book: Szabo, A.; Ostlund, N. S., *Modern Quantum Chemistry: Introduction to Advanced Electronic Structure Theory*. Dover Publications, Inc.: Mineola, New York, 1996

Q: What parameters are in B3LYP?

Parameters in B3LYP

Parameters (in red) in B3LYP:

$$E_{XC} = (1 - a)E_{\mathrm{X}}^{\mathrm{LDA}} + aE_{\mathrm{X}}^{\mathrm{HF}} + bE_{\mathrm{X}}^{\mathrm{GGA}} + cE_{\mathrm{C}}^{\mathrm{GGA}} + (1 - c)E_{\mathrm{C}}^{\mathrm{LDA}}$$

How this parameters were obtained?
They were optimized on the G2 data set with atomization energies, ionization potentials, proton affinities and total atomic energies (a = 0.20, b = 0.72, and c = 0.81).

B3LYP: A. D. Becke. *J. Chem. Phys.* **1993,** *98*, 1372.
Book: W. Koch, M. C. Holthausen, *A Chemist's Guide to Density Functional Theory*. Second ed.; WILEY-VCH Verlag GmbH: Weinheim, **2001**; pp. 293

ML algorithms

Some of the ML Algorithms

- Various types of neural networks (NN), **deep learning**

- Gaussian processes (GP)
- Kernel ridge regression (KRR)
- Support vector machines (SVMs) & support vector regression (SVR)

- Linear regression!
- Decision trees
- k-Nearest neighbor algorithm
- and many more…

Parametric vs nonparametric algorithms

Some of the ML Algorithms

f(x; **parameters**)

Linear regression

$$f(\mathbf{x}_i; \boldsymbol{\beta}) = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots$$

Number of parameters is fixed: parametric model

Neural networks are also parametric models

Kernel ridge regression (KRR)

$$f(\mathbf{x}_i; \mathbf{p}) = \sum_{j=1}^{N_{\mathrm{tr}}} \alpha_j k(\mathbf{x}_i, \mathbf{x}_j; \mathbf{b})$$

Number of parameters depends on number of training points: nonparametric model, e.g. KRR

ML Algorithms

- Various types of neural networks (NN), **deep learning**

- Gaussian processes (GP)
- Kernel ridge regression (KRR)
- Support vector machines (SVMs) & support vector regression (SVR)

- Linear regression!
- Decision trees
- k-Nearest neighbor algorithm
- and many more…

ML algorithms

Linear regression

$$f(\mathbf{x}_i;\boldsymbol{\beta}) = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots$$

Kernel ridge regression (KRR)

Neural networks (NN)

ML algorithms

Linear regression

$$f(\mathbf{x}_i;\boldsymbol{\beta}) = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots$$

Kernel ridge regression (KRR)

Neural networks (NN)

- Literature:
  T. Hastie, R. Tibshirani, J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed.; Springer-Verlag: New York, **2009**; pp. 763.
- M. Rupp. *Int. J. Quantum Chem.* **2015,** *115*, 1058–1073
- T. Hofmann, B. Schölkopf, A. J. Smola. *Ann. Statist.* **2008,** *36*, 1171
- P. O. Dral. *J. Comput. Chem.* **2019,** *40*, 2339
- P. O. Dral, Quantum Chemistry Assisted by Machine Learning. In *Advances in Quantum Chemistry: Chemical Physics and Quantum Chemistry*, 1st ed.; Ruud, K.; Brändas, E. J., Eds. Academic Press: **2020**; Vol. 81, pp. 291

Linear regression

Multiple linear regression

$$f(\mathbf{x}_i;\boldsymbol{\beta}) = \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots$$

$$f(\mathbf{x}_i;\boldsymbol{\beta}) = \sum_{j=1}^{p} \beta_j x_{ij}$$

$$f(\mathbf{x}_i;\boldsymbol{\beta}) = \mathbf{x}_i^T \boldsymbol{\beta}$$

How to find the coefficients β_j?

Linear regression

Multiple linear regression

$$f(\mathbf{x}_i; \boldsymbol{\beta}) = \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots$$

$$f(\mathbf{x}_i; \boldsymbol{\beta}) = \sum_{j=1}^{p} \beta_j x_{ij}$$

$$f(\mathbf{x}_i; \boldsymbol{\beta}) = \mathbf{x}_i^T \boldsymbol{\beta}$$

We can find the coefficients $\boldsymbol{\beta}$ using the method of least squares, where coefficients are fit to get the minimum residual sum of squares (RSS) with respect to the training set with N_{tr} reference values $\mathbf{y}$:

$$\arg\min_{\boldsymbol{\beta}} \sum_{i=1}^{N_{tr}} (f(\mathbf{x}_i; \boldsymbol{\beta}) - y_i)^2$$

Linear regression

$$\arg\min_{\boldsymbol{\beta}} \sum_{i=1}^{N_{tr}} (f(\mathbf{x}_i; \boldsymbol{\beta}) - y_i)^2$$

$$L(\boldsymbol{\beta}) = \sum_{i=1}^{N_{tr}} \left(\mathbf{x}_i^T \boldsymbol{\beta} - y_i\right)^2$$

$$L(\boldsymbol{\beta}) = (\mathbf{X}\boldsymbol{\beta} - \mathbf{y})^T (\mathbf{X}\boldsymbol{\beta} - \mathbf{y})$$

$$\mathbf{X} = \begin{pmatrix} x_{11} & \cdots & x_{1p} \\ \vdots & \ddots & \vdots \\ x_{N_{\mathrm{tr}}1} & \cdots & x_{N_{\mathrm{tr}}p} \end{pmatrix}$$

$$\frac{\partial L(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = 2\mathbf{X}^T (\mathbf{X}\boldsymbol{\beta} - \mathbf{y}) = \mathbf{0}$$

Linear regression

$$\mathbf{X}^T\mathbf{X}\boldsymbol{\beta} = \mathbf{X}^T\mathbf{y}$$

$$\boldsymbol{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

Linear regression has an analytical solution!

While it is very advantageous, it assumes that the data follow the linear distribution, which is often not the case

Linear regression

Task 4: Fit linear model $E = aR$ on a training set with 20 points sampled along H_2 dissociation curve (energies E in Hartree at FCI/aug-cc-pV6Z; internuclear distances R in Angstrom)

Calculate R^2 and residual sum of squares (RSS)

Task 5: Fit linear regression with intercept *b*?

Calculate R^2 and residual sum of squares (RSS)

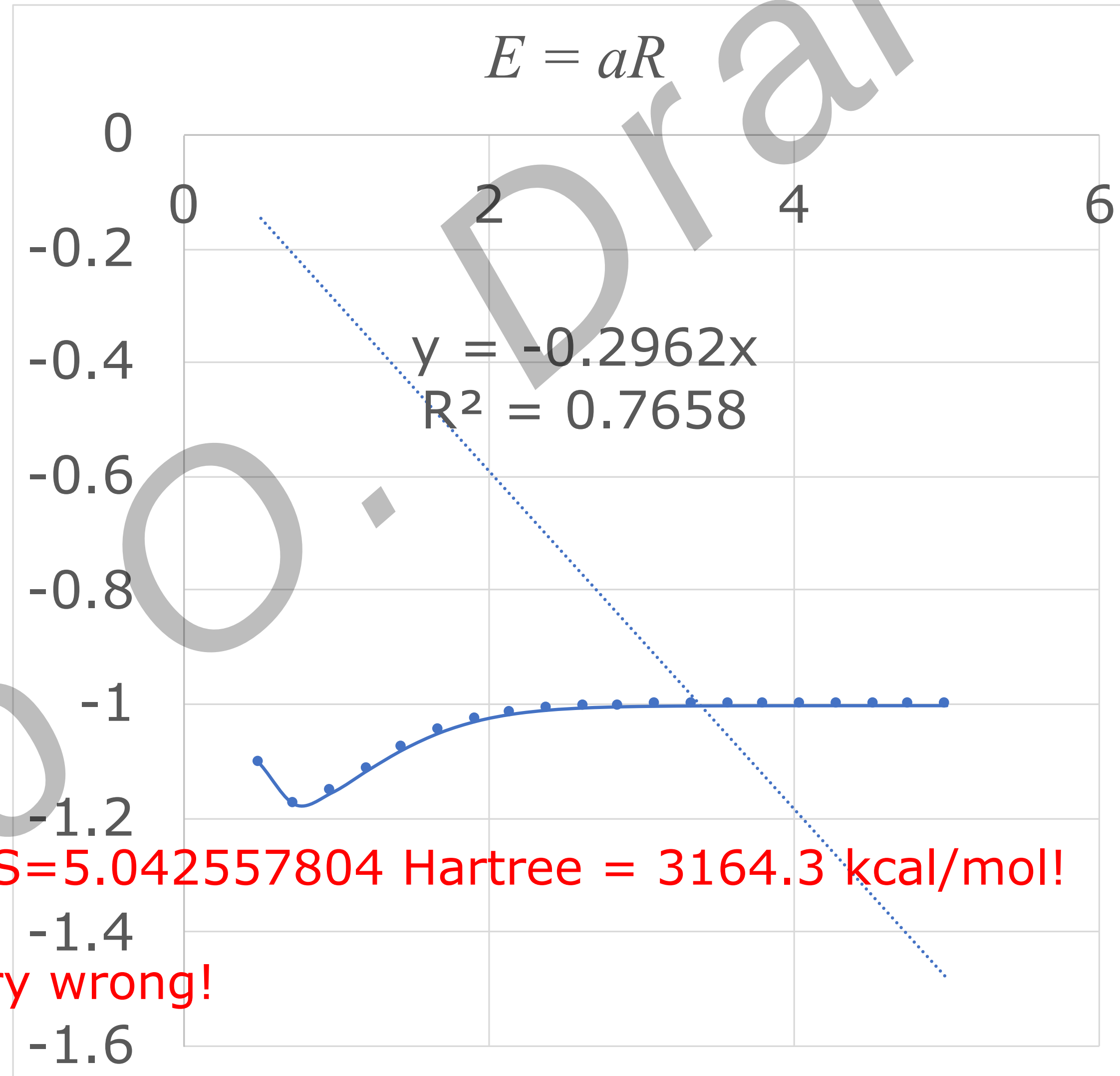
$$E = aR+b$$

$$x_{i1} = R$$

$$\mathrm{RSS} = \sum_{i=1}^{N_{tr}} \left(f(\mathbf{x}_i; \boldsymbol{\beta}) - y_i\right)^2$$

Linear regression

H_2 dissociation curve



RSS=5.042557804 Hartree = 3164.3 kcal/mol!

Very wrong!

Linear regression

Task 5: Fit linear regression with intercept *b*?

Calculate R^2 and residual sum of squares (RSS)

$$E = aR + b$$

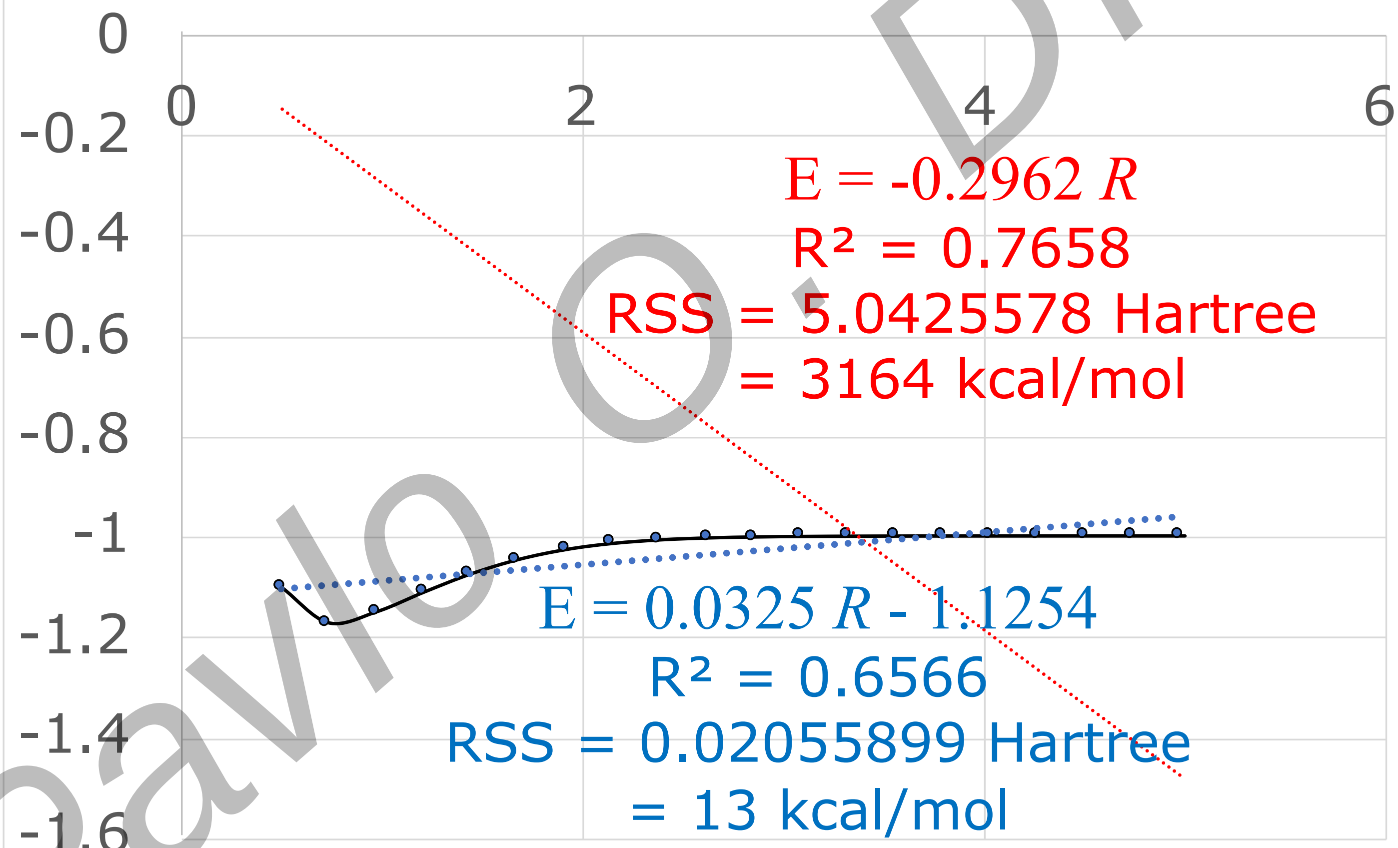
$$\mathrm{RSS} = \sum_{i=1}^{N_{tr}} \left(f(\mathbf{x}_i; \boldsymbol{\beta}) - y_i \right)^2$$

Linear regression

Much better,
but still qualitatively wrong!

$$E = aR$$

$$E = aR + b$$



Linear regression

Q: What about linear regression with intercept *b*?

$$E = aR+b$$

It is equivalent to mapping function $R \rightarrow (R, 1)$, $\Phi\big((R)\big) = (R, 1)$, where Φ maps from *p*-dimensional input space into p^d-dimensional feature space
Now we can solve multiple linear regression with two variables

$$f(\mathbf{x}_i; \boldsymbol{\beta}) = \beta_1 x_{i1} + \beta_2 x_{i2} = \beta_1 R + \beta_2 1 = aR + b$$

$$x_{i1} = R_i$$

$$x_{i2} = 1$$

$$\beta_1 = a$$

$$\beta_2 = b$$

Linear regression

Any ideas how to get the dissociation curve shape right?

Linear regression

Q: Any ideas how to get the dissociation curve shape right?

We can use mapping $R \rightarrow (R^{-6}, R^{-12}, 1)$ inspired by Lennard-Jones potential, this allows us to treat data set (*E*, *R*) nonlinear in input space (*R*) using vectors in feature space $(R^{-6}, R^{-12}, 1)$

Now we can solve multiple linear regression with three variables:

$$f(\mathbf{x}_i; \boldsymbol{\beta}) = \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3} = \beta_1 R_i^{-6} + \beta_2 R_i^{-12} + \beta_3 1 = aR_i^{-6} + bR_i^{-12} + c$$

$$x_{i1} = R_i^{-6}$$

$$x_{i2} = R_i^{-12}$$

$$x_{i3} = 1$$

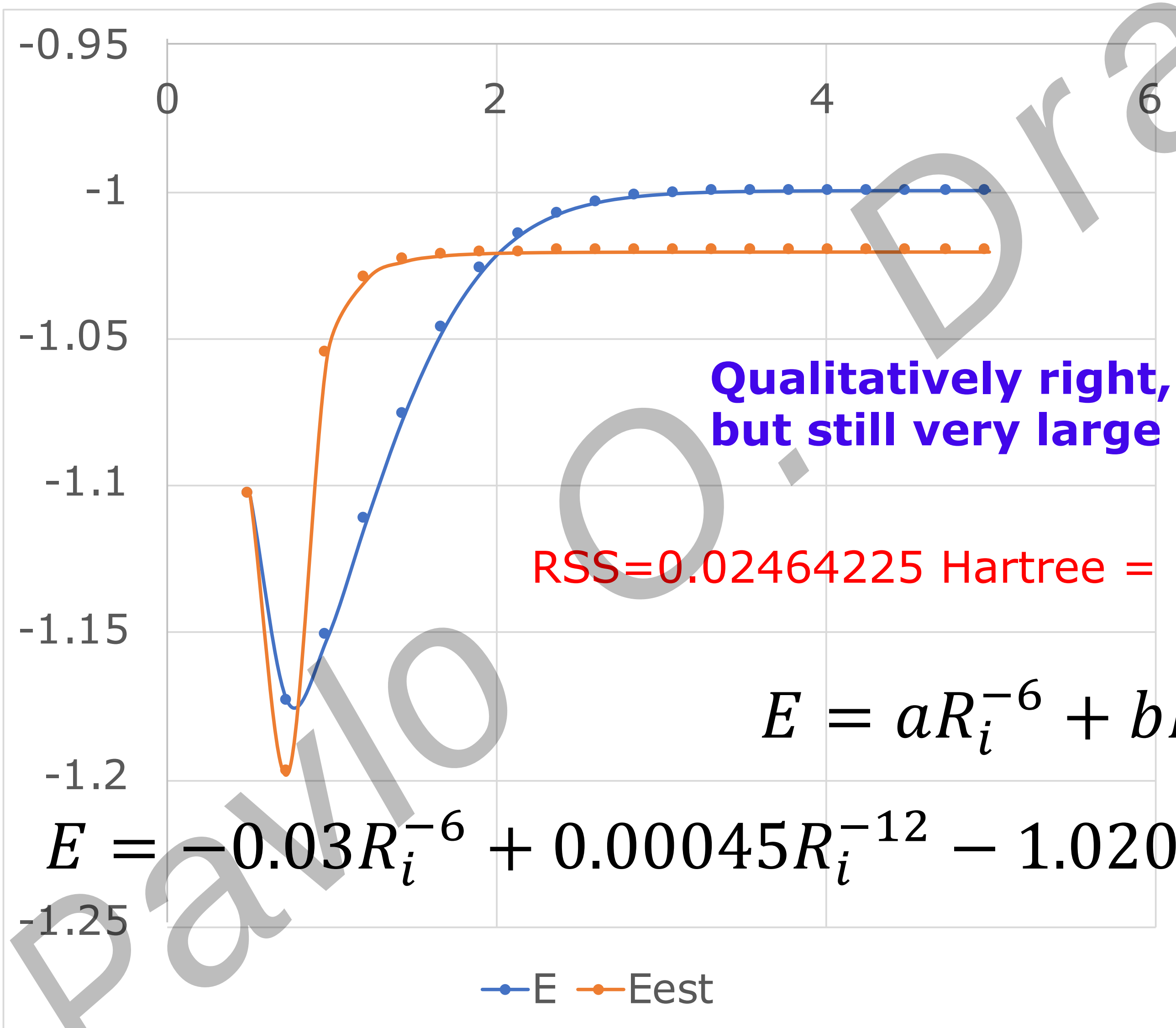
$$\beta_1 = a$$

$$\beta_2 = b$$

$$\beta_3 = c$$

Max Pinheiro Jr, P. O. Dral, Kernel methods.
In *Quantum Chemistry in the Age of Machine Learning*, P. O. Dral, Ed. Elsevier: **2022**, *in press*.
Paperback ISBN: 9780323900492

Linear regression



Qualitatively right,
but still very large error!

RSS=0.02464225 Hartree = 15 kcal/mol!

$$E = aR_i^{-6} + bR_i^{-12} + c$$

$$E = -0.03R_i^{-6} + 0.00045R_i^{-12} - 1.02060806$$

Linear regression

Can we extend it to more variables and make it more flexible?

Yes! We can go to infinite number of variables!

How?

Using a kernel trick

Linear regression

Let's rewrite the linear regression equation by representing the regression coefficients via a sum over all training points:

$$f(\mathbf{x}';\boldsymbol{\beta}) = \sum_{j=1}^{p} \beta_j x'_j$$

$$\beta_j = \sum_{i=1}^{N_{tr}} \alpha_i x_{ij}$$

$$f(\mathbf{x}') = \sum_{j=1}^{p}\left(\sum_{i=1}^{N_{tr}} \alpha_i x_{ij}\right) x'_j = \sum_{i=1}^{N_{tr}} \alpha_i \sum_{j=1}^{p} x_{ij} x'_j = \sum_{i=1}^{N_{tr}} \alpha_i \mathbf{x}_i^T \mathbf{x}'$$

$\mathbf{x}_i^T \mathbf{x}' = \langle \mathbf{x}_i, \mathbf{x}' \rangle$ Dot-product = inner product = scalar product of two vectors

Linear regression

As we have seen before, we can map vectors $\mathbf{x}$ and $\mathbf{x}'$ from p-dimensional input space into p^d-dimensional feature space using mapping function Φ:

$$f(\mathbf{x}') = \sum_{i=1}^{N_{tr}} \alpha_i \Phi(\mathbf{x}_i)^T \Phi(\mathbf{x}')$$

In previous examples we new the mapping function and explicit forms of vectors in the feature space. But all we need is a dot-product between vectors in the feature space, not their explicit forms. Such dot-product is called ***kernel*** denoted $k(\mathbf{x}_i, \mathbf{x}')$ and it is calculated in using vectors in the input space (not feature space!):

$$k(\mathbf{x}_i, \mathbf{x}') = \Phi(\mathbf{x}_i)^T \Phi(\mathbf{x}')$$

The kernel trick is substitution of the calculation of the dot-product using explicit representations of vectors in the feature space by using a kernel function:

$$f(\mathbf{x}') = \sum_{i=1}^{N_{tr}} \alpha_i k(\mathbf{x}_i, \mathbf{x}')$$

Kernel-based machine learning

$$f(\mathbf{x}') = \sum_{i=1}^{N_{tr}} \alpha_i k(\mathbf{x}_i, \mathbf{x}')$$

This is a kernel-based machine learning function.

Kernel trick allows us to use tools of linear regression for data nonlinear in the input space by converting variables into (higher dimensional) feature space.

Q: How to find the regression coefficients α?

Kernel-based machine learning

$$f(\mathbf{x}') = \sum_{i=1}^{N_{tr}} \alpha_i k(\mathbf{x}_i, \mathbf{x}')$$

This is a kernel-based machine learning function.

Kernel trick allows us to use tools of linear regression for data nonlinear in the input space by converting variables into (higher dimensional) feature space.

We can find the coefficients $\boldsymbol{\alpha}$ using the method of least squares, where coefficients are fit to get the minimum residual sum of squares (RSS) with respect to the training set with N_{tr} reference values **y**:

$$\arg\min_{\boldsymbol{\alpha}} \sum_{i=1}^{N_{tr}} (f(\mathbf{x}_i; \boldsymbol{\alpha}) - y_i)^2$$

Kernel-based machine learning

$$\arg\min_{\boldsymbol{\alpha}} \sum_{i=1}^{N_{tr}} (f(\mathbf{x}_i; \boldsymbol{\alpha}) - y_i)^2$$

$$L(\boldsymbol{\beta}) = \sum_{i=1}^{N_{tr}} (f(\mathbf{x}_i; \boldsymbol{\alpha}) - y_i)^2$$

$$f(\mathbf{x}_i) = \sum_{j=1}^{N_{tr}} \alpha_j k(\mathbf{x}_i, \mathbf{x}_j)$$

$$L(\boldsymbol{\beta}) = \sum_{i=1}^{N_{tr}} \left(\sum_{j=1}^{N_{tr}} \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) - y_i \right)^2$$

$$L(\boldsymbol{\beta}) = (\mathbf{K}\boldsymbol{\alpha} - \mathbf{y})^T (\mathbf{K}\boldsymbol{\alpha} - \mathbf{y})$$

$$\mathbf{K} = \begin{pmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & \cdots & k\left(\mathbf{x}_1, \mathbf{x}_{N_{\mathrm{tr}}}\right) \\ \vdots & \ddots & \vdots \\ k\left(\mathbf{x}_{N_{\mathrm{tr}}}, \mathbf{x}_1\right) & \cdots & k\left(\mathbf{x}_{N_{\mathrm{tr}}}, \mathbf{x}_{N_{\mathrm{tr}}}\right) \end{pmatrix}$$

Kernel matrix

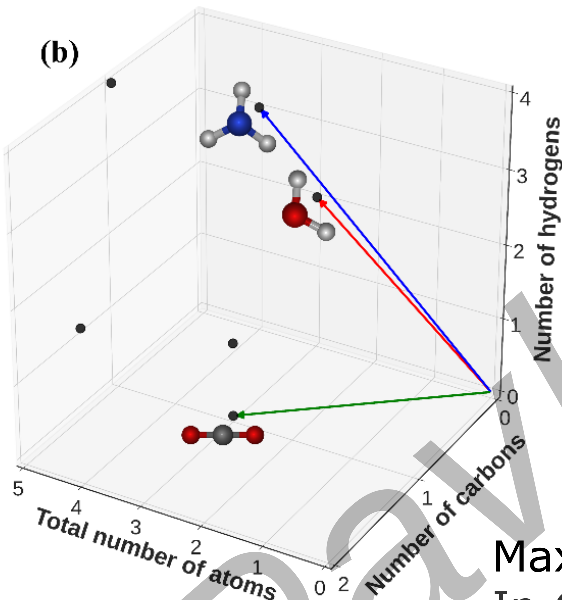
Kernel-based machine learning

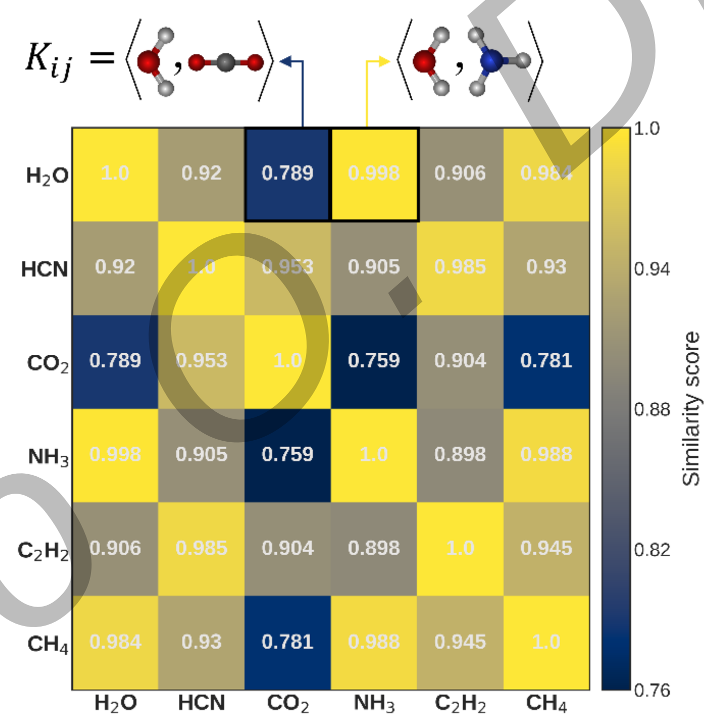
Kernel matrix
(matrix measuring similarities between points – here cosine)

$$\cos\theta = \mathbf{a}\cdot\mathbf{b}/(\|\mathbf{a}\|\|\mathbf{b}\|)$$

(a)

Molecule	num_atoms	num_carbons	num_hydrogens
H2O	3	0	2
HCN	3	1	1
CO2	3	1	0
NH3	4	0	3
C2H2	4	2	2
CH4	5	1	4





Max Pinheiro Jr, P. O. Dral, Kernel methods.
In *Quantum Chemistry in the Age of Machine Learning*, P. O. Dral, Ed. Elsevier: **2022**, *in press*.
Paperback ISBN: 9780323900492

Kernel-based machine learning

$$\arg\min_{\boldsymbol{\alpha}} \sum_{i=1}^{N_{tr}} (f(\mathbf{x}_i; \boldsymbol{\alpha}) - y_i)^2$$

$$L(\boldsymbol{\beta}) = (\mathbf{K}\boldsymbol{\alpha} - \mathbf{y})^T (\mathbf{K}\boldsymbol{\alpha} - \mathbf{y})$$

$$\frac{\partial L(\boldsymbol{\alpha})}{\partial \boldsymbol{\alpha}} = \mathbf{2K}^T(\mathbf{K}\boldsymbol{\alpha} - \mathbf{y}) = \mathbf{2K}(\mathbf{K}\boldsymbol{\alpha} - \mathbf{y}) = \mathbf{2KK}\boldsymbol{\alpha} - \mathbf{2Ky} = \mathbf{0}$$

$$\mathbf{KK}\boldsymbol{\alpha} = \mathbf{Ky}$$

$$\mathbf{K}^{-1}\mathbf{KK}\boldsymbol{\alpha} = \mathbf{K}^{-1}\mathbf{Ky}$$

$$\mathbf{K}\boldsymbol{\alpha} = \mathbf{y}$$

$$\boldsymbol{\alpha} = \mathbf{K}^{-1}\mathbf{y}$$

Kernel-based ML vs linear regression

$$f(\mathbf{x}') = \sum_{i=1}^{N_{tr}} \alpha_i k(\mathbf{x}_i, \mathbf{x}')$$

This is a kernel-based machine learning function.

If the kernel function is itself a dot-product:

$$k(\mathbf{x}_i, \mathbf{x}') = \mathbf{x}_i^T \mathbf{x}'$$

The expression becomes equivalent to the linear regression as we have seen above and that is why such a dot-product kernel is also called "linear kernel":

$$f(\mathbf{x}') = \sum_{i=1}^{N_{tr}} \alpha_i k(\mathbf{x}_i, \mathbf{x}') = \sum_{i=1}^{N_{tr}} \alpha_i \mathbf{x}_i^T \mathbf{x}' = \sum_{j=1}^{p} \beta_j x'_j$$

$$\beta_j = \sum_{i=1}^{N_{tr}} \alpha_i x_{ij}$$

Kernel-based machine learning

$$\beta_j = \sum_{i=1}^{N_{tr}} \alpha_i x_{ij}$$

One can get the same β_j for infinite combinations of α_i

Some solutions will have very large α_i with opposite signs trying to compensate each other

$$\beta = \sum_{i=1}^{N_{\mathrm{tr}}=20} \alpha_i x_i = \sum_{i=1}^{N_{\mathrm{tr}}=20} \alpha_i R_i$$

$$\beta = (-0.2962 \cdot 2) \cdot 0.5 + \sum_{i=2}^{N_{\mathrm{tr}}=20} 0 x_i$$

$$\beta = 198476910439 \cdot 0.5 - 19847691043.95924 \cdot 5 + \sum_{i=3}^{N_{\mathrm{tr}}=20} 0 x_i$$

$$f(x;\beta) = \beta x = -0.2962x$$

$$\boldsymbol{\alpha} = \mathbf{K}^{-1}\mathbf{y}$$

Prone to overfitting, numerically unstable

Often, K is not invertible matrix

Shrinkage methods

Ridge regression – belongs to shrinkage methods (useful for feature importance analysis)

$$\arg\min_{\boldsymbol{\beta}} (\mathbf{X}\boldsymbol{\beta} - \mathbf{y})^T (\mathbf{X}\boldsymbol{\beta} - \mathbf{y}) + \lambda \boldsymbol{\beta}^T \boldsymbol{\beta}$$

$$\boldsymbol{\beta} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}$$

Another example of shrinkage method is the lasso

$$\arg\min_{\boldsymbol{\beta}} (\mathbf{X}\boldsymbol{\beta} - \mathbf{y})^T (\mathbf{X}\boldsymbol{\beta} - \mathbf{y}) + \lambda \sum_{i=1}^{p} |\beta_i|$$

Identity matrix

$$\mathbf{I} = \begin{pmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 1 \end{pmatrix}$$

Kernel ridge regression (KRR)

$$\arg\min_{\boldsymbol{\alpha}} (\mathbf{K}\boldsymbol{\alpha} - \mathbf{y})^T (\mathbf{K}\boldsymbol{\alpha} - \mathbf{y}) + \lambda \boldsymbol{\alpha}^T \mathbf{K}\boldsymbol{\alpha}$$

$$\boldsymbol{\alpha} = (\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{y}$$

Coefficient magnitude is forced to shrunk with larger λ in these methods

λ is nonnegative regularization hyperparameter, smoothens function and makes solution numerically more stable.

Kernel-based ML with Gaussian kernel

$$f(\mathbf{x}') = \sum_{i=1}^{N_{\text{tr}}} \alpha_i k(\mathbf{x}_i, \mathbf{x}')$$

This is a kernel-based machine learning function.

One of the popular kernel functions is the Gaussian kernel function:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{1}{2\sigma^2}\sum_{s}^{N_x}\left(x_{i,s} - x_{j,s}\right)^2\right)$$

It maps vectors **x** from N_x-dimensional input space into infinite-dimensional feature space.

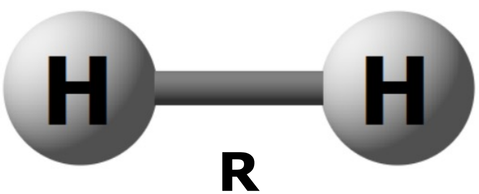
σ is a positive hyperparameter defining the length scale of the Gaussian function.

Model Selection

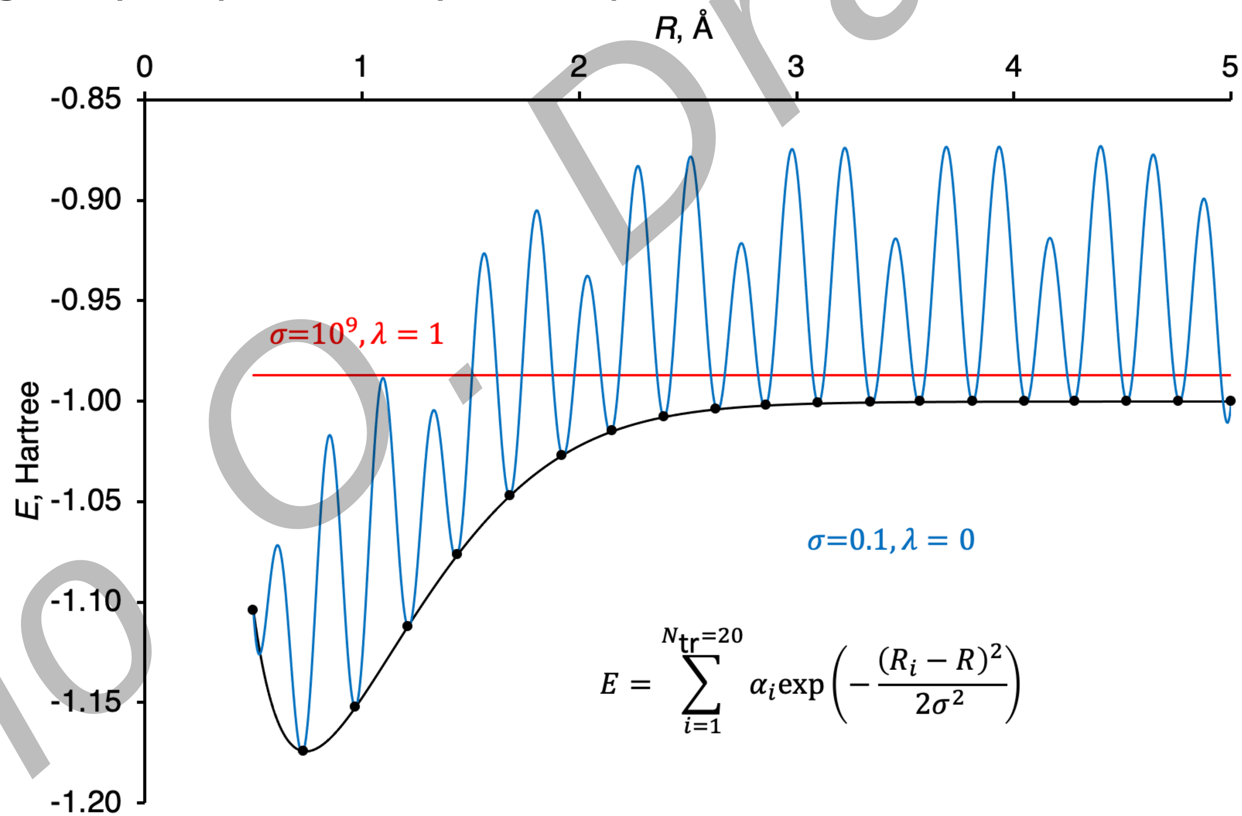
Many ML algorithms can give you practically ideal predictions for their own training set

Underfitting
Low variance
High bias

H_2 dissociation curve

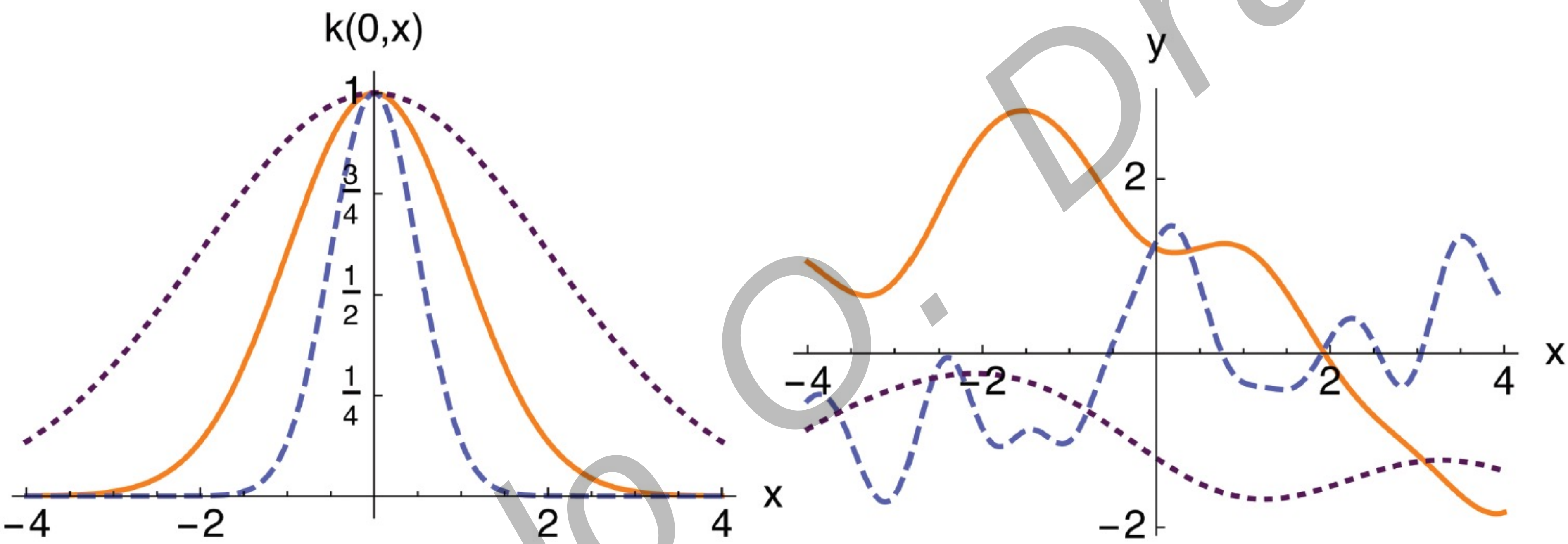


Overfitting
High variance
Low bias



Max Pinheiro Jr, P. O. Dral, Kernel methods. In *Quantum Chemistry in the Age of Machine Learning*, P. O. Dral, Ed. Elsevier: **2022**, *in press*.
Paperback ISBN: 9780323900492

KRR with Gaussian kernel



(b) Gaussian kernel with $\sigma = 0.5, 1, 2$ (dashed, solid, dotted lines).

Figure from: M. Rupp. *Int. J. Quantum Chem.* **2015,** *115*, 1058

KRR with Gaussian kernel: H_2

Take KRR with Gaussian kernel

$$f(\mathbf{x}') = \sum_{i=1}^{N_{tr}} \alpha_i \exp\left(-\frac{1}{2\sigma^2}\sum_{s}^{N_x}\left(x_{i,s} - x'_s\right)^2\right)$$

and consider what happens for very small $\sigma \to 0$:

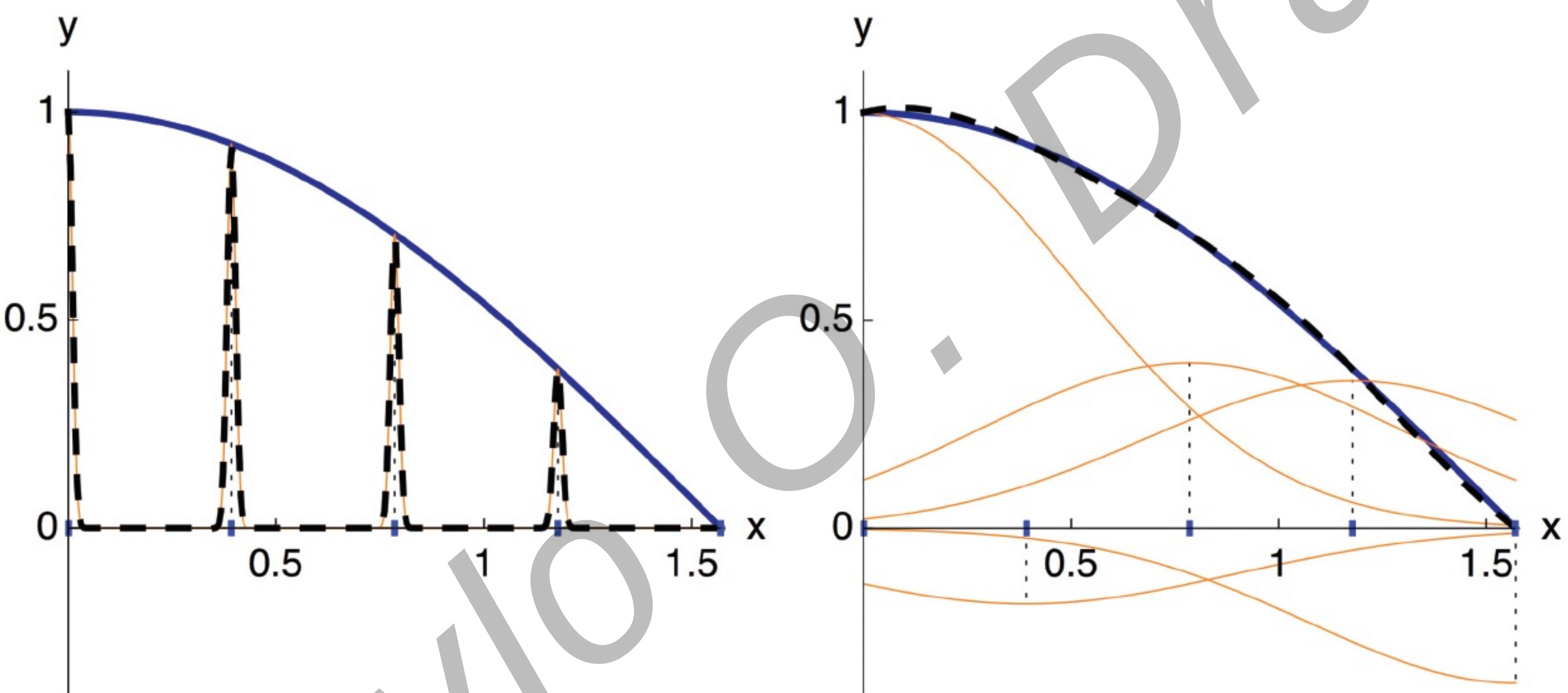
$$f(\mathbf{x}') = \begin{cases} \alpha_i, \text{for } \mathbf{x}' = \mathbf{x}_i \\ 0, \text{for } \mathbf{x}' \neq \mathbf{x}_i \end{cases}$$

and consider what happens for very large $\sigma \to \infty$:

$$f(\mathbf{x}') = \sum_{i=1}^{N_{tr}} \alpha_i = const$$

P. O. Dral, Quantum Chemistry Assisted by Machine Learning. In *Advances in Quantum Chemistry: Chemical Physics and Quantum Chemistry Volume 81*, 1st ed.; Brandas, E.; Ruud, K., Eds. Academic Press: 2020; Vol. 81. **Online tutorial:** MLatom.com/AQCtutorial/

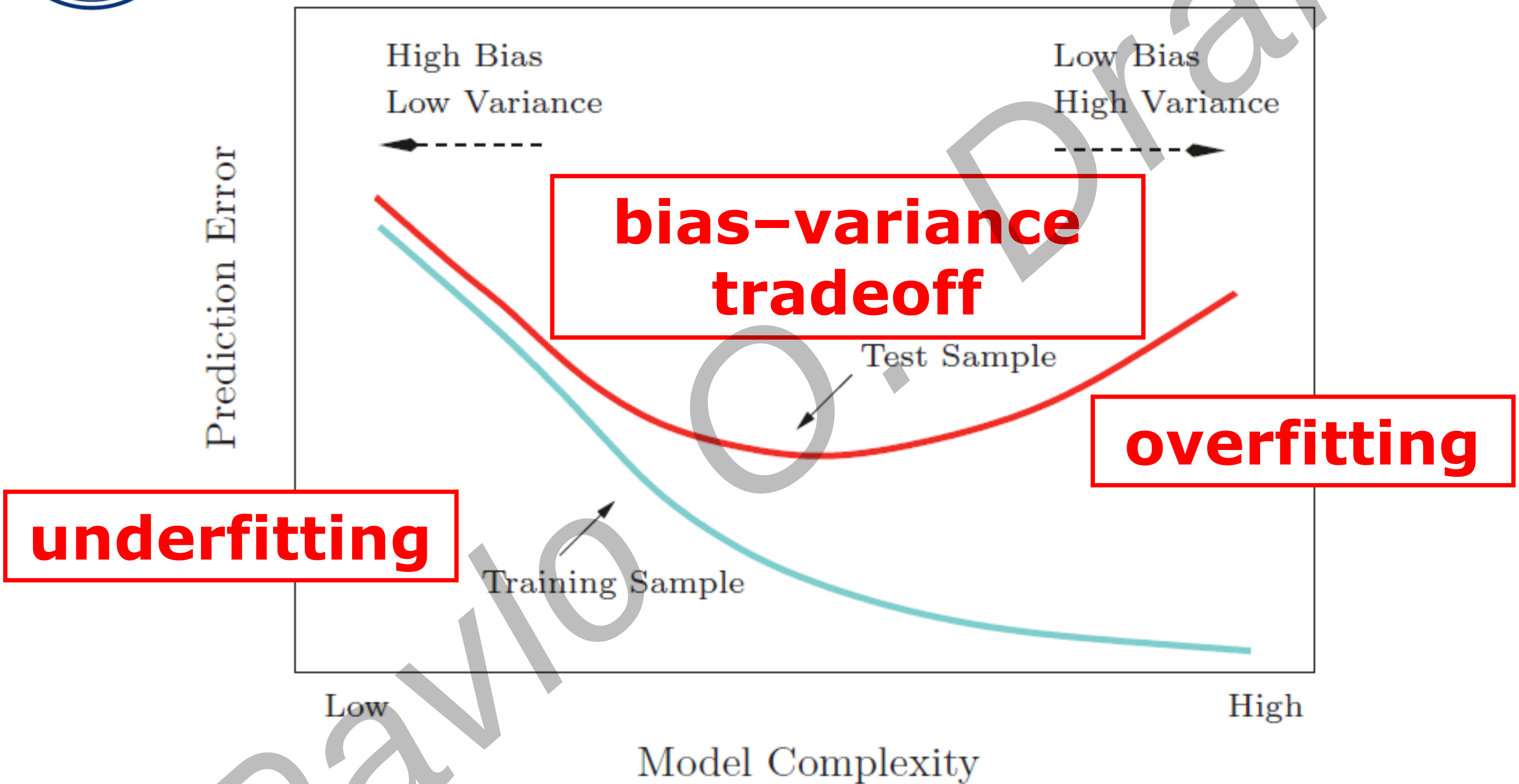
KRR with Gaussian kernel



(a) Overfitting ($\sigma = 0.01$)   (b) Fitting ($\sigma = 0.5$)

Figure from: M. Rupp. *Int. J. Quantum Chem.* **2015,** *115*, 1058

Model Selection



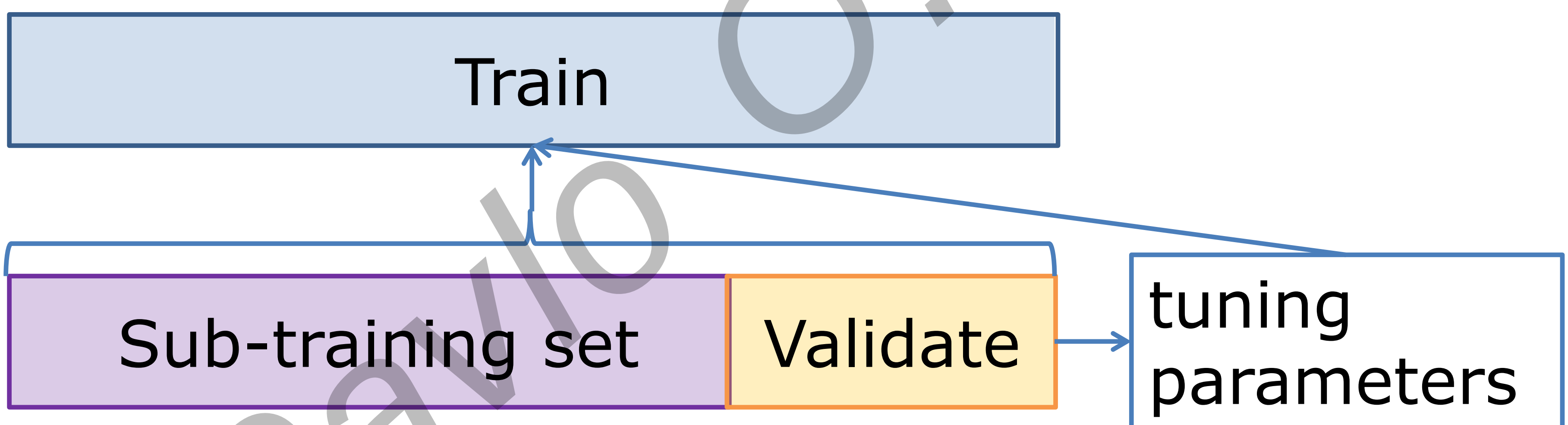
Hastie, Tibshirani, Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed.; Springer-Verlag, **2009**

Model selection (hyperparameter tuning)

Q: How to choose hyperparameters?

Model selection (hyperparameter tuning)

We target minimal error **not** in the training set, but in the validation set for models trained on the sub-training set.

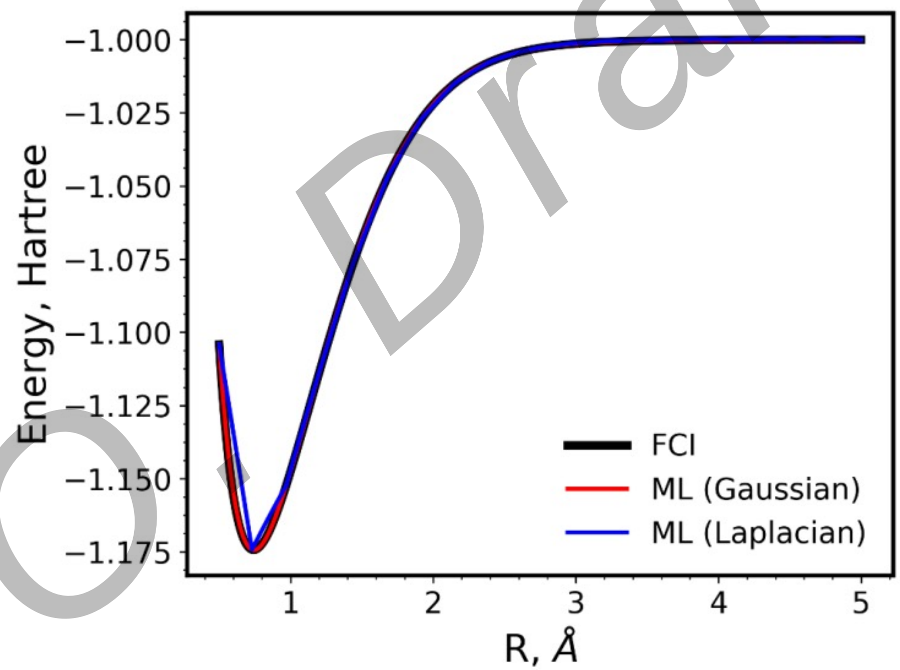


Hastie, Tibshirani, Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed.; Springer-Verlag, **2009**

Kernel-based model with Gaussian kernel

H_2 dissociation curve

Full CI calculations: more than 30 min for one value of R.



ML trained on 20 points needs less than 1 sec. for hundreds of other points

P. O. Dral, Quantum Chemistry Assisted by Machine Learning. In *Advances in Quantum Chemistry: Chemical Physics and Quantum Chemistry Volume 81*, 1st ed.; Brandas, E.; Ruud, K., Eds. Academic Press: 2020; Vol. 81. **Online tutorial:** MLatom.com/AQCtutorial/

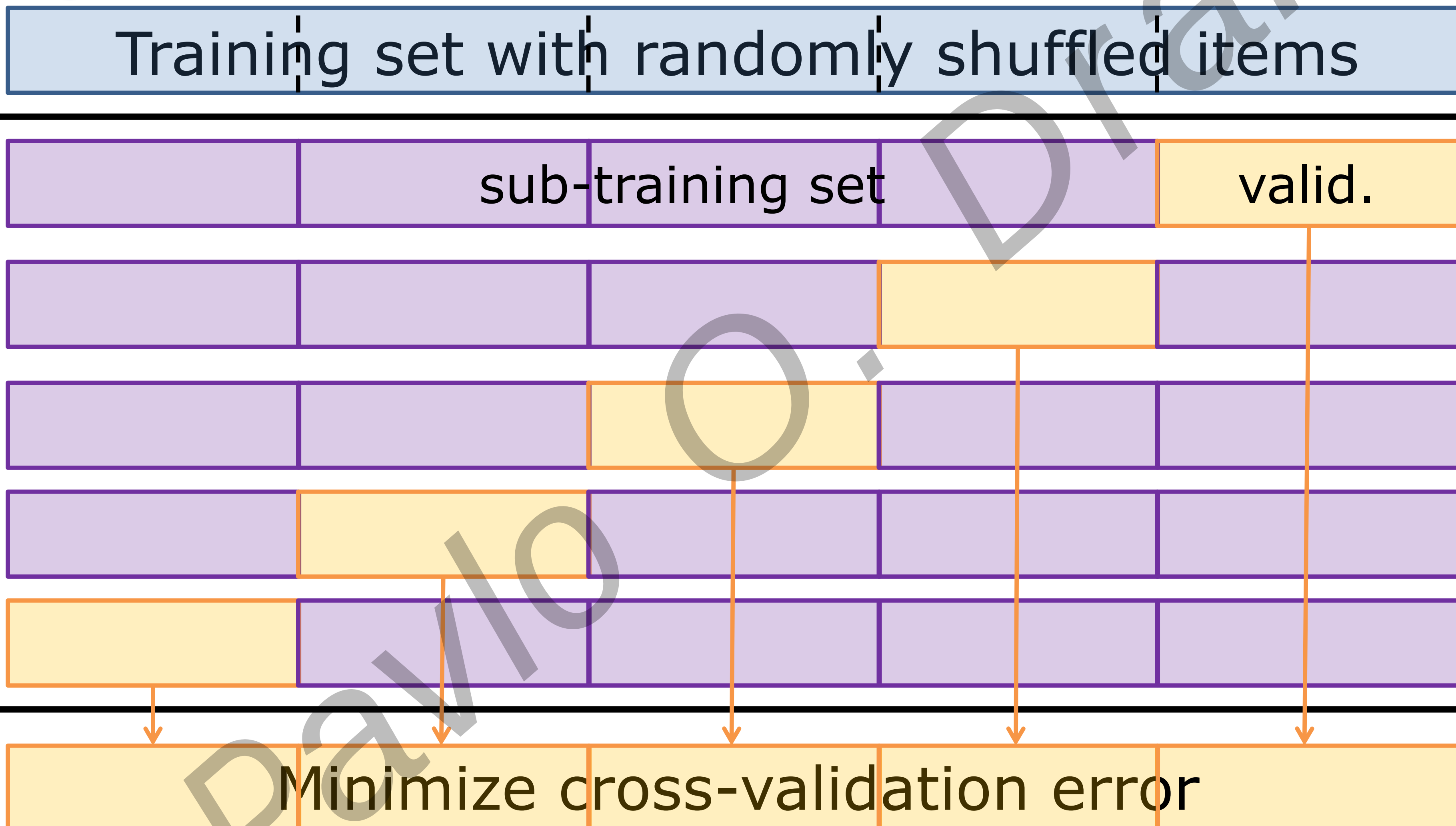
5-fold Cross-validation

Training set with randomly shuffled items

5-fold Cross-validation

Training set with randomly shuffled items				
sub-training set				valid.

5-fold Cross-validation



Model Selection

Random sampling for model selection is not always a good idea

Sometimes, **stratification** is preferable

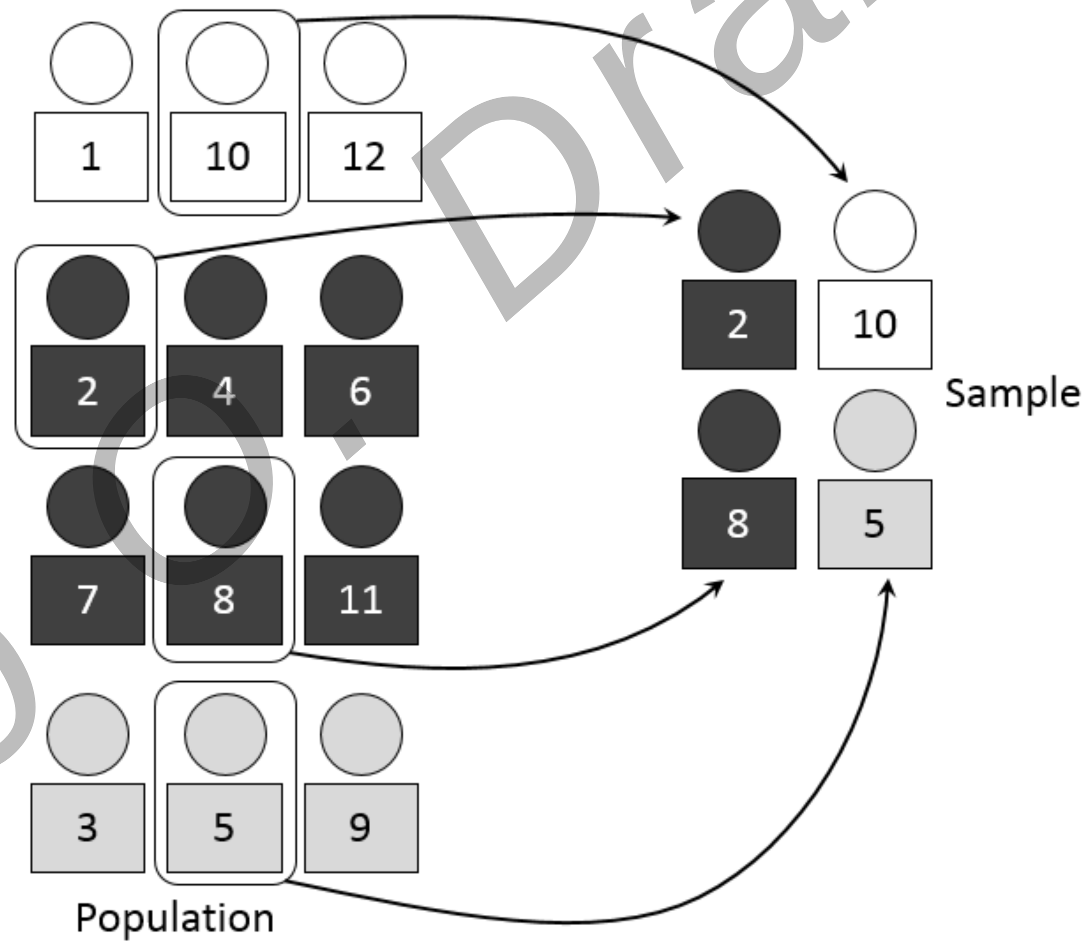


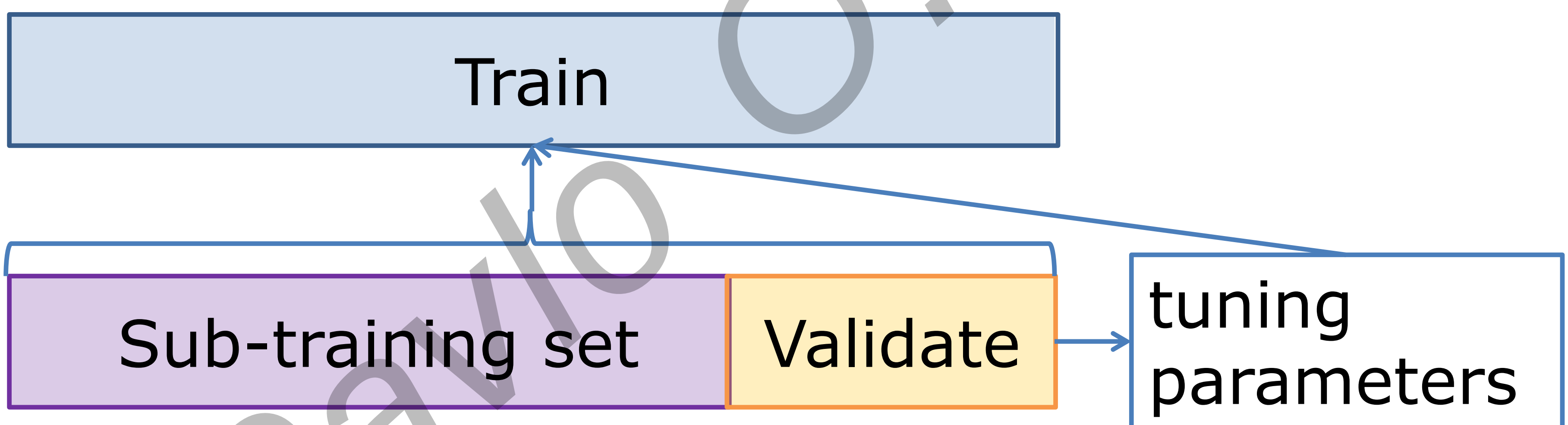
Figure by Dan Kernler [CC BY-SA 4.0], from Wikimedia Commons

Model Evaluation

(estimation of the generalization error)

ML: Error Estimation

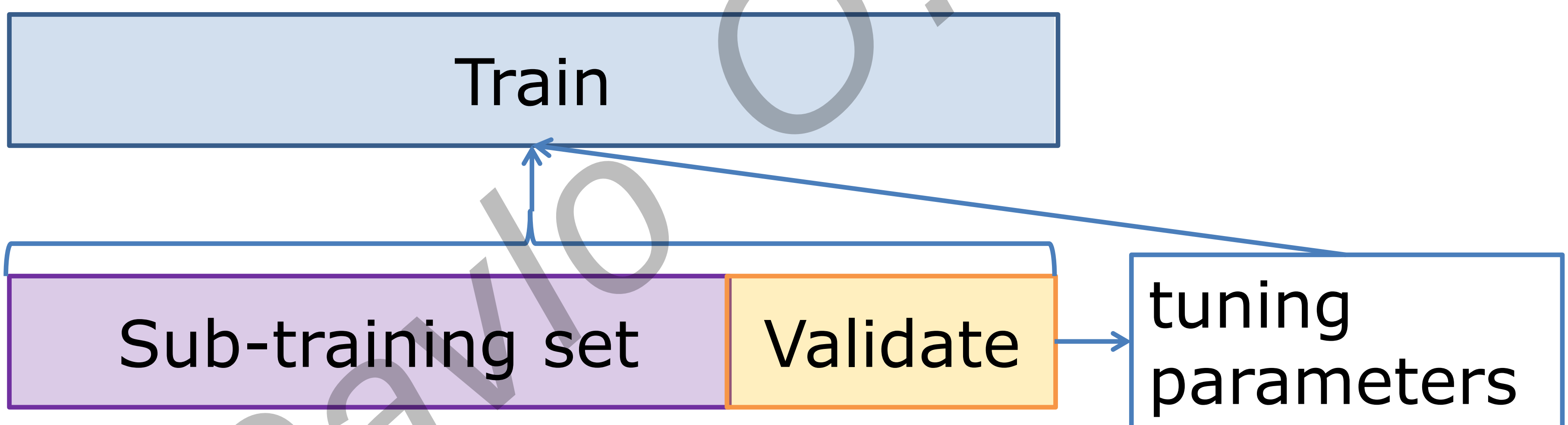
- Often ML error for its own training set is close to zero



Hastie, Tibshirani, Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed.; Springer-Verlag, **2009**

ML: Error Estimation

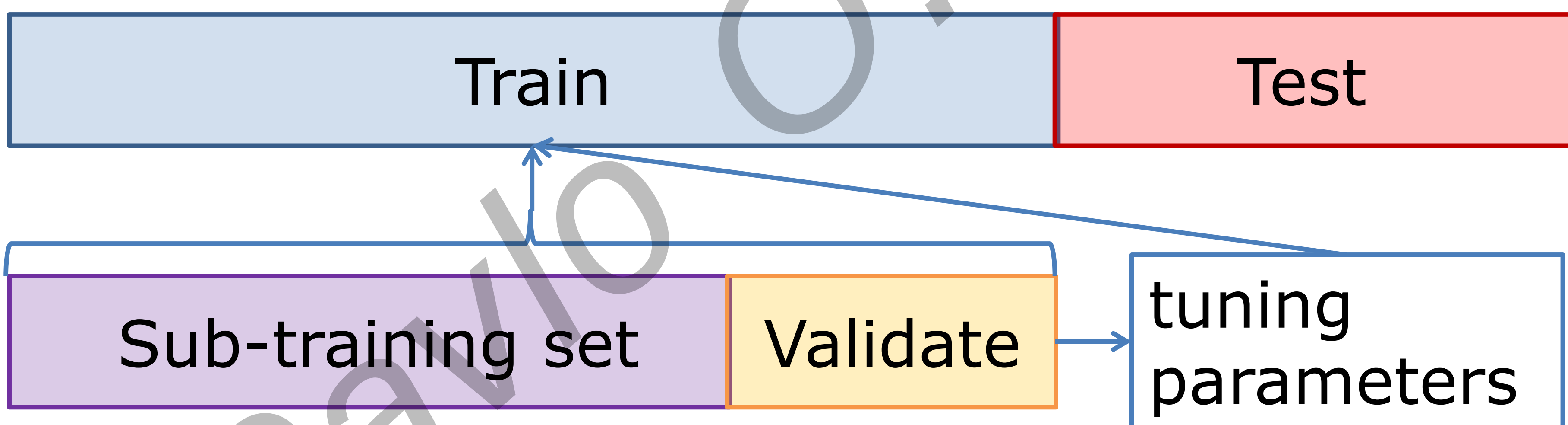
- Often ML error for its own training set is close to zero
- Using errors in the validation set would be also incorrect, because their minimization is a part of the training process



Hastie, Tibshirani, Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed.; Springer-Verlag, **2009**

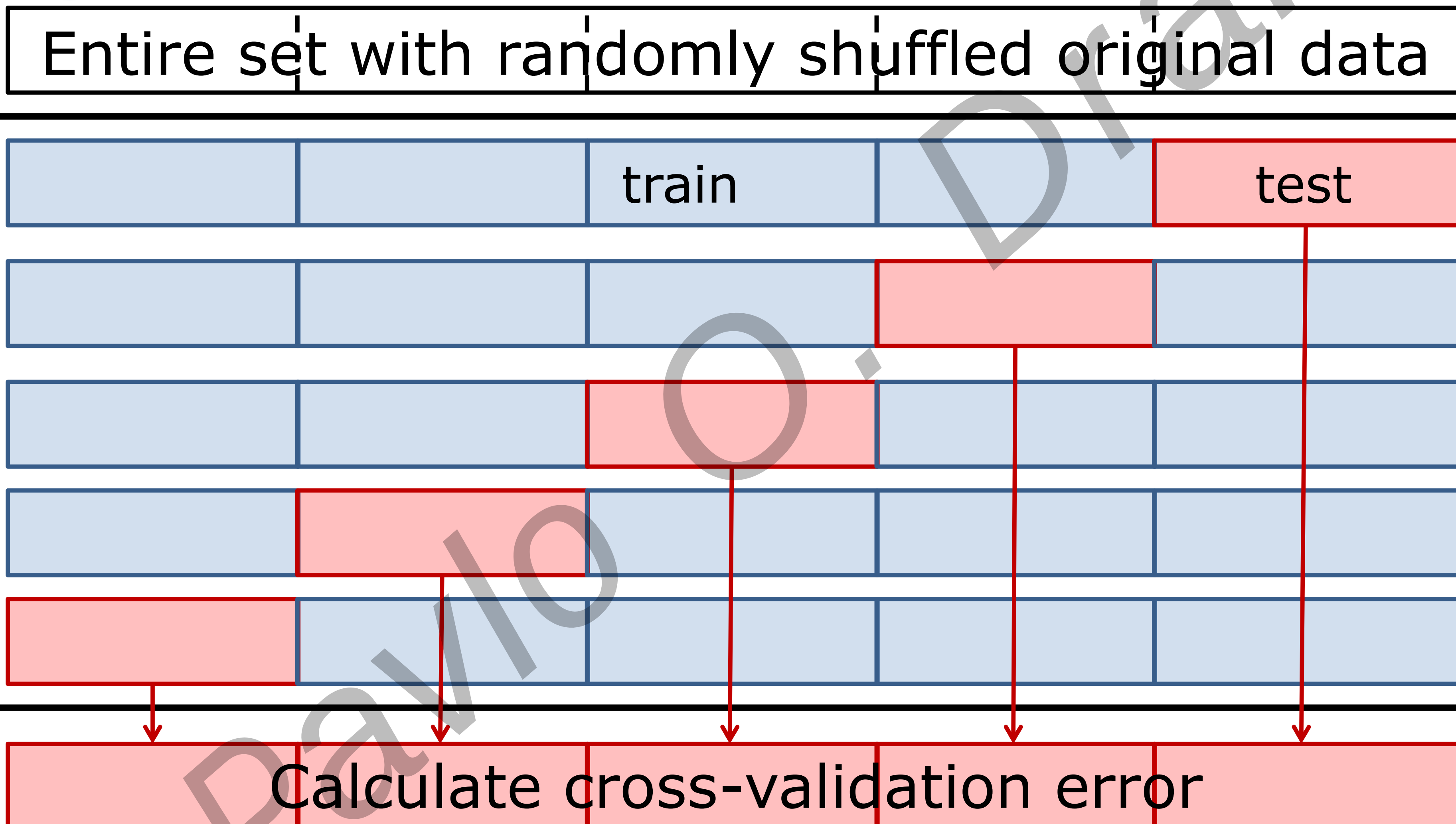
ML: Error Estimation

- Often ML error for its own training set is close to zero
- Using errors in the validation set would be also incorrect, because their minimization is a part of the training process
- We should estimate errors on a **completely independent test set**

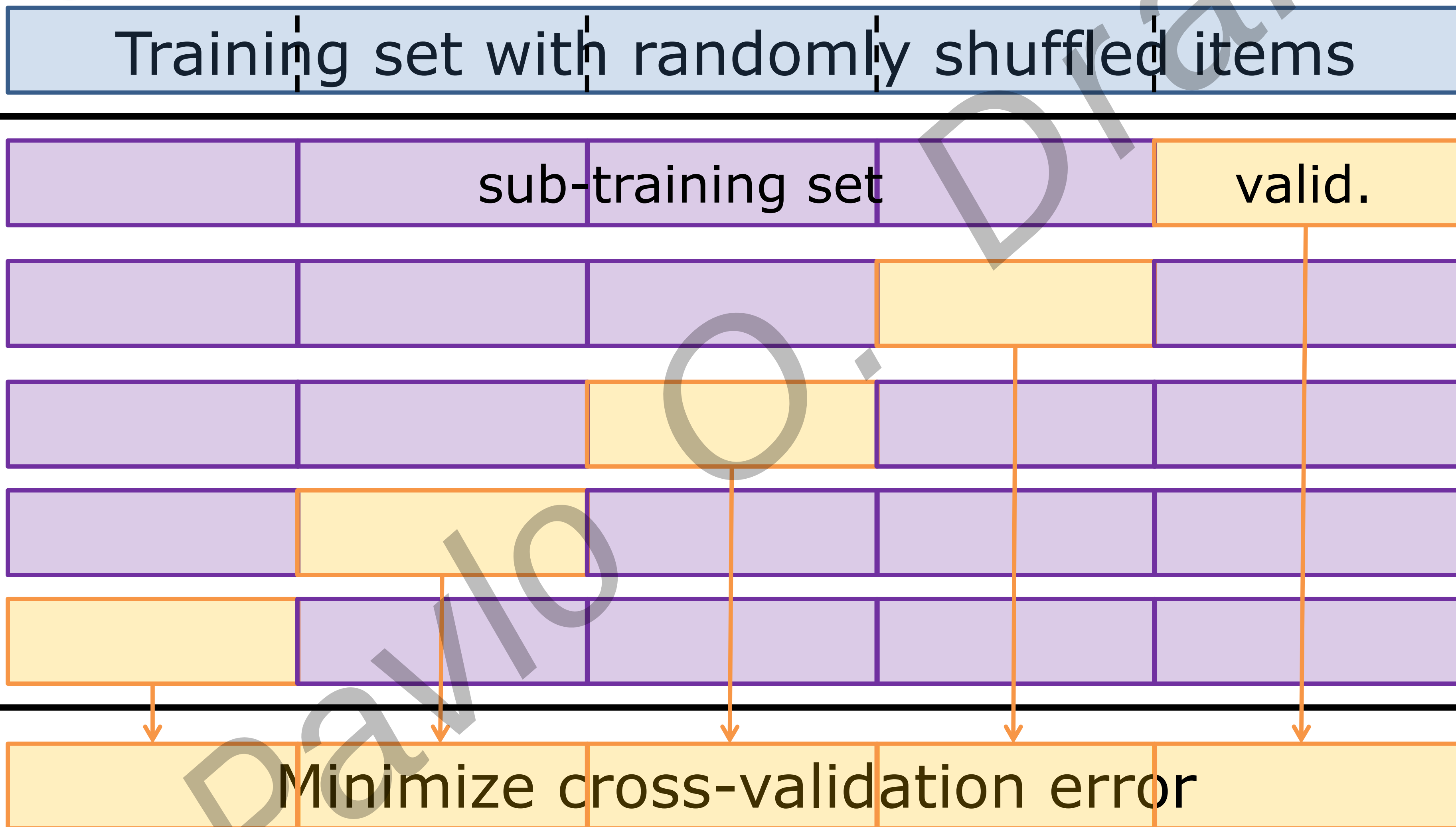


Hastie, Tibshirani, Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed.; Springer-Verlag, **2009**

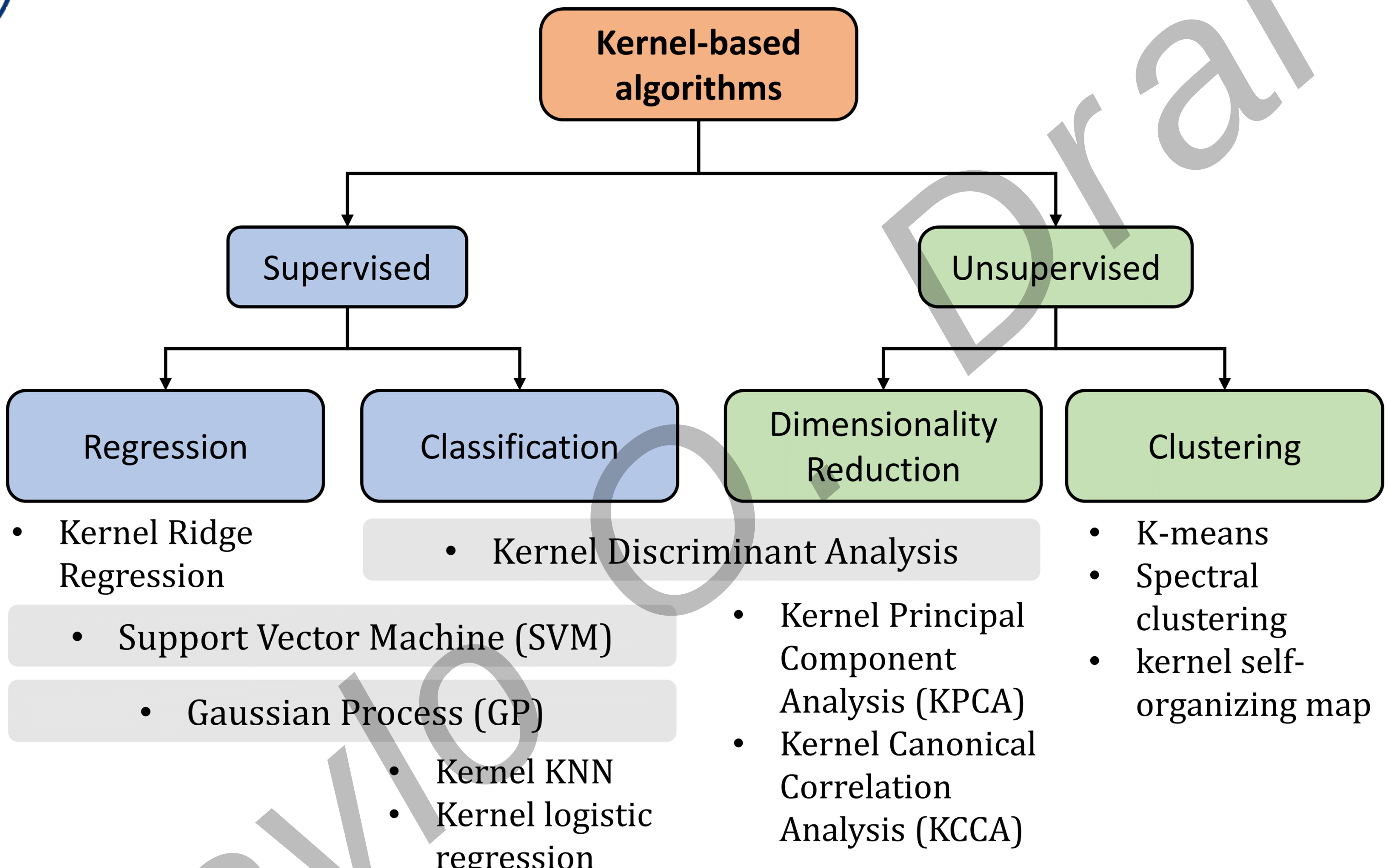
5-fold Cross-validation



5-fold Cross-validation



Family of kernel methods



Max Pinheiro Jr, P. O. Dral, Kernel methods. In *Quantum Chemistry in the Age of Machine Learning*, P. O. Dral, Ed. Elsevier: **2022**, *in press*.
Paperback ISBN: 9780323900492

KRR and other kernel methods

- Kernel ridge regression (KRR)

$$f(\mathbf{x}') = \sum_{i=1}^{N_{tr}} \alpha_i k(\mathbf{x}_i, \mathbf{x}')$$

- Gaussian processes (GP, kriging)

$$f(\mathbf{x}') = \sum_{i=1}^{N_{tr}} \alpha_i k(\mathbf{x}_i, \mathbf{x}')$$

Prediction functions are the same for KRR and GP!

- Support vector machines (SVM)

$$f(\mathbf{x}') = \sum_{i=1}^{N_{tr}} \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}'),\ 0 < \alpha_i < C$$

See, for example:

- T. Hastie, R. Tibshirani, J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed.; Springer-Verlag: New York, **2009**; pp. 763
- Rasmussen, Williams, *Gaussian Processes for Machine Learning*. The MIT Press: Boston, **2006**

KRR and other kernel methods

- Kernel ridge regression gives the same prediction as Gaussian processes:

$$f(\mathbf{x}') = \sum_{i=1}^{N_{tr}} \alpha_i k(\mathbf{x}_i, \mathbf{x}')$$

- Gaussian processes also provide:
  - variance V

$$\mathbf{k}' = \begin{pmatrix} k(\mathbf{x}_1, \mathbf{x}') \\ \vdots \\ k\left(\mathbf{x}_{N_{\mathrm{tr}}}, \mathbf{x}'\right) \end{pmatrix}$$

$$V(\mathbf{x}') = k(\mathbf{x}', \mathbf{x}') - \mathbf{k}'^T(\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{k}'$$

  - Marginal likelihood:

$$\log p(\mathrm{y}|X) = -\frac{1}{2}\mathbf{y}^T\boldsymbol{\alpha} - \frac{1}{2}\log|\mathbf{K} + \lambda\mathbf{I}| - \frac{N_{tr}}{2}\log 2\pi$$

Hyperparameters in kernel function can be found by optimizing log marginal likelihood, for which derivatives are taken, e.g. $\frac{\partial \log p(\mathrm{y}|X, \sigma)}{\partial \sigma}$

Rasmussen, Williams, *Gaussian Processes for Machine Learning*. The MIT Press: Boston, **2006**

Pros & cons of kernel methods

Advantages of kernel methods:
- Nonparametric models, i.e., do not assume a specific behavior of data (compare to parametric model such as linear regression)
- Explicitly incorporate training data, thus very flexible and accurate
- Closed (analytical) solution, i.e. fast training

$$f(\mathbf{x}') = \sum_{i=1}^{N_{tr}} \alpha_i k(\mathbf{x}_i, \mathbf{x}')$$

$$\boldsymbol{\alpha} = (\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{y}$$

Disadvantages:
- Slow training for lots of training data (scales as $O(N_{tr}^3)$)
- Requires lots of RAM to store the kernel matrix (scales as $O(N_{tr}^2)$)
- Prediction time slows down with more training data (scales as $O(N_{tr}^1)$)

P. O. Dral, Quantum Chemistry Assisted by Machine Learning. In *Advances in Quantum Chemistry: Chemical Physics and Quantum Chemistry Volume 81*, 1st ed.; Brandas, E.; Ruud, K., Eds. Academic Press: 2020; Vol. 81. **Online tutorial:** MLatom.com/AQCtutorial/

Pros & cons of kernel methods

Table 1 Required memory (RAM) for storing kernel matrix built with increasing number of training points.

Training set size	RAM size
$100 = 10^2$	78 kB
$1000 = 10^3$	7.6 MB
$10{,}000 = 10^4$	0.75 GB
$50{,}000 = 5 \times 10^4$	19 GB
$100{,}000 = 10^5$	75 GB
$500{,}000 = 5 \times 10^5$	1.8 TB
$1000{,}000 = 10^6$	7.3 TB

Table 2 CPU time needed for calculating regression coefficients for increasing number of training points, assuming that it takes 10 s for 10,000 training points.

Training set size	Time
$100 = 10^2$	0.01 milliseconds
$1000 = 10^3$	0.01 s
$10{,}000 = 10^4$	10 s
$50{,}000 = 5 \times 10^4$	21 min
$100{,}000 = 10^5$	2.8 h
$500{,}000 = 5 \times 10^5$	15 days
$1000{,}000 = 10^6$	3.9 months

P. O. Dral, Quantum Chemistry Assisted by Machine Learning. In *Advances in Quantum Chemistry: Chemical Physics and Quantum Chemistry Volume 81*, 1st ed.; Brandas, E.; Ruud, K., Eds. Academic Press: 2020; Vol. 81. **Online tutorial:** MLatom.com/AQCtutorial/

Kernel Ridge Regression

Solutions:

- Reduce the training set by selecting the most relevant points[1,2]
- Sparsification techniques[3]
- Construct high-dimensional kernels as products of one-dimensional kernels[4]

See, for example:
[1] Dral, Owens, Yurchenko, Thiel, *J. Chem. Phys.* **2017**, *146*, 244108
[2] Hu, Xie, Li, Li, Lan, *J. Phys. Chem. Lett.* **2018**, *9*, 2725
[3] Bartók, Csányi, *Int. J. Quantum Chem.* **2015**, *115*, 1051
[4] Unke, Meuwly, *J. Chem. Inf. Model.* **2017**, *57*, 1923

Neural networks

ML algorithms

Linear regression

$$f(\mathbf{x}_i; \boldsymbol{\beta}) = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots$$

Kernel ridge regression (KRR)

Neural networks (NN)

ML algorithms

Linear regression

$$f(\mathbf{x}_i; \boldsymbol{\beta}) = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots$$

Kernel ridge regression (KRR)

Neural networks (NN)

Literature:

- P. O. Dral, A. Kananenka, F. Ge, B.-X. Xue, Neural Networks. In *Quantum Chemistry in the Age of Machine Learning*, 1st ed.; P. O. Dral, Ed. Elsevier: 2022. In press.
- T. Hastie, R. Tibshirani, J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed.; Springer-Verlag: New York, **2009**; pp. 763.
- K. T. Schütt, S. Chmiela, O. A. von Lilienfeld, A. Tkatchenko, K. Tsuda, K.-R. Müller, Machine Learning Meets Quantum Physics. Springer: Cham, 2020.
- I. Goodfellow, Y. Bengio, A. Courville, *Deep Learning*. MIT Press: **2016**. http://www.deeplearningbook.org

ML algorithms

Linear regression

$$\hat{y} = f(\mathbf{x}; \mathbf{w}, b) = b + w_1 x_1 + w_2 x_2 + \cdots + w_p x_p = \mathbf{x}^T \mathbf{w} + b$$

$$\downarrow$$

Neural networks (NNs): the single hidden layer, feed-forward network

$$\hat{y} = f(\mathbf{x}; \boldsymbol{\alpha}, \mathbf{a}, \mathbf{w}, b) = b + w_1 h_1(\mathbf{x}; \boldsymbol{\alpha}_1, a_1) + \cdots + w_M h_M(\mathbf{x}; \boldsymbol{\alpha}_M, a_M) = \mathbf{h}^T \mathbf{w} + b$$

$$h_m(\mathbf{x}; \boldsymbol{\alpha}_m, a_m) = g\left(a_m + \alpha_{m1} x_1 + \alpha_{m2} x_2 + \cdots + \alpha_{mp} x_p\right) = g(\mathbf{x}^T \boldsymbol{\alpha}_m + a_m)$$

$$\hat{y} = f(\mathbf{x}; \boldsymbol{\alpha}, \mathbf{a}, \mathbf{w}, b) = f^{(2)}(\mathbf{h}; \mathbf{w}, b) = f^{(2)}\left(f^{(1)}(\mathbf{x})\right)$$

Activation functions:

$$g(v) = \exp(-a(v-c)^2)$$ radial basis function (RBF)

P. O. Dral, A. Kananenka, F. Ge, B.-X. Xue, Neural Networks. In *Quantum Chemistry in the Age of Machine Learning*, 1st ed.; P. O. Dral, Ed. Elsevier: 2022. In press.

NNs

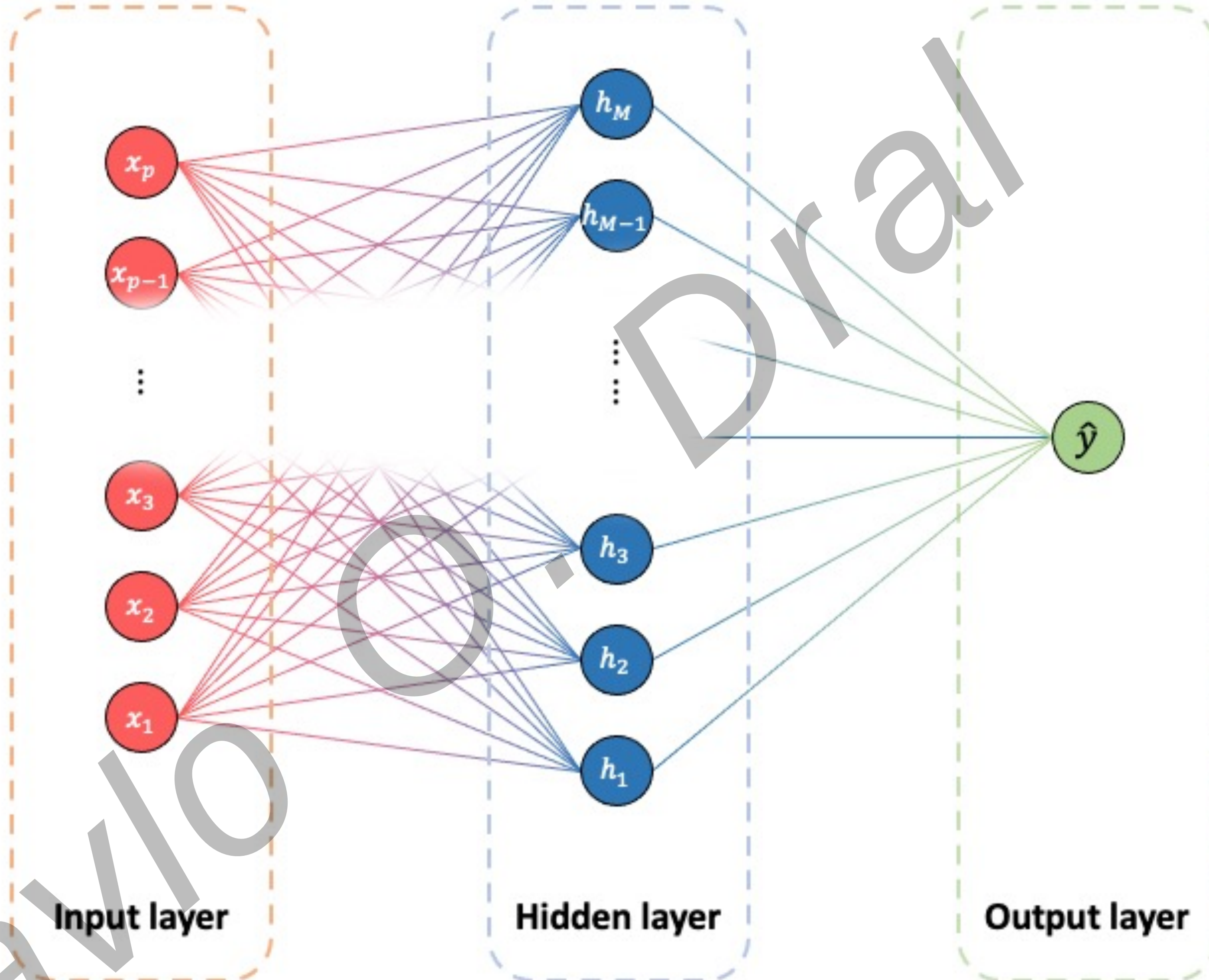
NN consists of
- layers
- nodes=units= neurons

The single hidden layer, feed-forward NN

$\mathbf{w}, \boldsymbol{\alpha}$ … weights

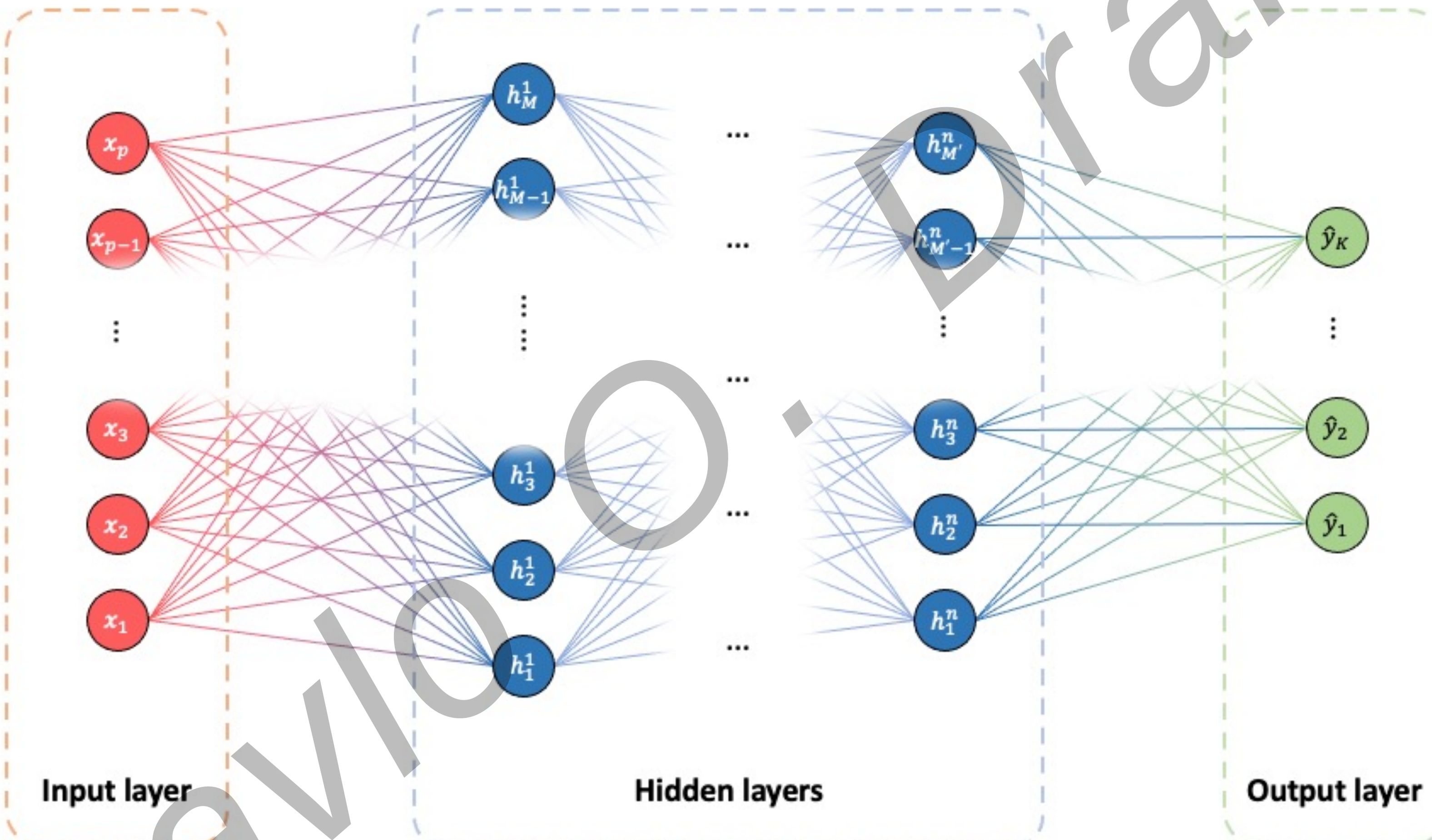
$\mathbf{a}$, b… biases

$$f(\mathbf{x}; \boldsymbol{\alpha}, \mathbf{a}, \mathbf{w}, b) = f^{(2)}(\mathbf{h}; \mathbf{w}, b)$$



P. O. Dral, A. Kananenka, F. Ge, B.-X. Xue, Neural Networks. In *Quantum Chemistry in the Age of Machine Learning*, 1st ed.; P. O. Dral, Ed. Elsevier: 2022. In press.

NNs



P. O. Dral, A. Kananenka, F. Ge, B.-X. Xue, Neural Networks. In *Quantum Chemistry in the Age of Machine Learning*, 1st ed.; P. O. Dral, Ed. Elsevier: 2022. In press.

ML algorithms

Linear regression

$$\hat{y} = f(\mathbf{x}; \mathbf{w}, b) = b + w_1 x_1 + w_2 x_2 + \cdots + w_p x_p = \mathbf{x}^T \mathbf{w} + b$$

$$\downarrow$$

Neural networks (NNs): the single hidden layer, feed-forward network

$$\hat{y} = f(\mathbf{x}; \boldsymbol{\alpha}, \mathbf{a}, \mathbf{w}, b) = b + w_1 h_1(\mathbf{x}; \boldsymbol{\alpha}_1, a_1) + \cdots + w_M h_M(\mathbf{x}; \boldsymbol{\alpha}_M, a_M) = \mathbf{h}^T \mathbf{w} + b$$

$$h_m(\mathbf{x}; \boldsymbol{\alpha}_m, a_m) = g\left(a_m + \alpha_{m1} x_1 + \alpha_{m2} x_2 + \cdots + \alpha_{mp} x_p\right) = g(\mathbf{x}^T \boldsymbol{\alpha}_m + a_m)$$

g is the activation function. If:

- g is the identity function, NN is equivalent to linear regression $g(v) = v$

$$g\left(a_m + \alpha_{m1} x_1 + \alpha_{m2} x_2 + \cdots + \alpha_{mp} x_p\right) = a_m + \alpha_{m1} x_1 + \alpha_{m2} x_2 + \cdots + \alpha_{mp} x_p$$

- Typically, g is used for the nonlinear transformation making the NN flexible

$g(v) = \exp(-a(v-c)^2)$ radial basis function (RBF)

P. O. Dral, A. Kananenka, F. Ge, B.-X. Xue, Neural Networks. In *Quantum Chemistry in the Age of Machine Learning*, 1st ed.; P. O. Dral, Ed. Elsevier: 2022. In press.

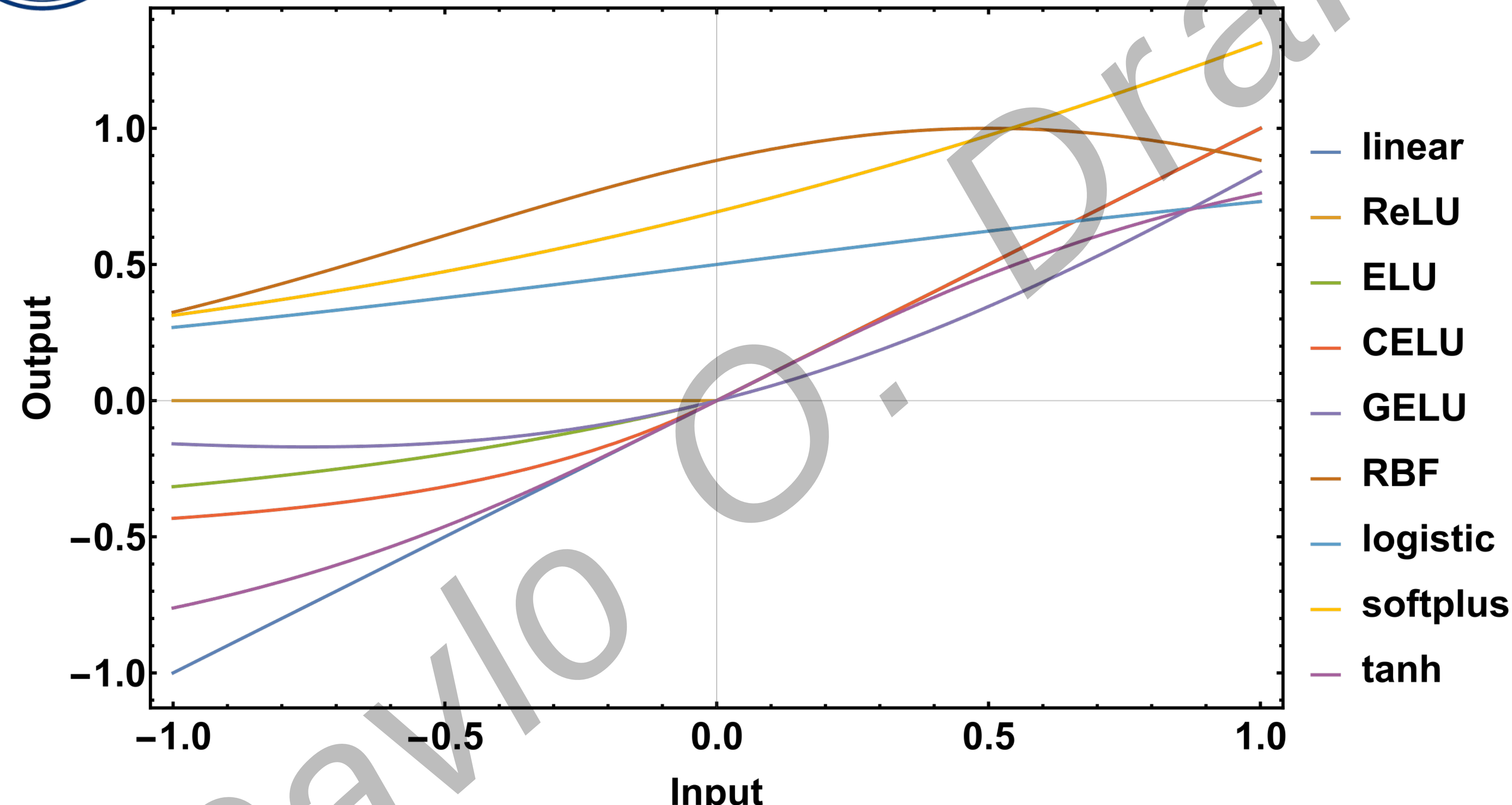
P. O. Dral, A. Kananenka, F. Ge, B.-X. Xue, Neural Networks. In *Quantum Chemistry in the Age of Machine Learning*, 1st ed.; P. O. Dral, Ed. Elsevier: 2022. In press.

Table 1. Overview of a selection of popular activation functions.

Names	Equation
linear function identity function[2]	$g(v) = v$
rectified linear unit (ReLU) [2]	$g(v) = \max(0, v)$
exponential linear unit (ELU)[5]	$g(v) = \begin{cases} v & \text{if } v \geq 0 \\ a(\exp(v) - 1) & \text{otherwise} \end{cases}$ where a is a parameter
continuously differentiable exponential linear unit (CELU)[6]	$g(v) = \begin{cases} v & \text{if } v \geq 0 \\ a\left(\exp\left(\frac{v}{a}\right) - 1\right) & \text{otherwise} \end{cases}$ where a is a parameter
Gaussian error linear unit (GELU)[7]	$g(v) = v \cdot \frac{1}{2}\left[1 + \operatorname{erf}\left(\frac{v}{\sqrt{2}}\right)\right]$ faster approximated versions: $g(v) = 0.5v\left(1 + \tanh\left[\sqrt{\frac{2}{\pi}}(v + 0.044715v^3)\right]\right)$ $g(v) = v \cdot \sigma(1.702v) = v \cdot \frac{1}{1 + \exp(-1.702v)}$

radial basis function (RBF)[1-2]	$g(v) = \exp(-a(v-c)^2)$ where a and c are parameters
logistic sigmoid function[1-2]	$g(v) = \sigma(v) = \dfrac{1}{1+\exp(-v)}$
softplus function[2]	$g(v) = \log(1+\exp(v))$
hyperbolic tangent function[2]	$g(v) = T(v) = \tanh(v)$

P. O. Dral, A. Kananenka, F. Ge, B.-X. Xue, Neural Networks. In *Quantum Chemistry in the Age of Machine Learning*, 1st ed.; P. O. Dral, Ed. Elsevier: 2022. In press.



P. O. Dral, A. Kananenka, F. Ge, B.-X. Xue, Neural Networks. In *Quantum Chemistry in the Age of Machine Learning*, 1st ed.; P. O. Dral, Ed. Elsevier: 2022. In press.

Training NNs

To train NN means to find its weights and biases

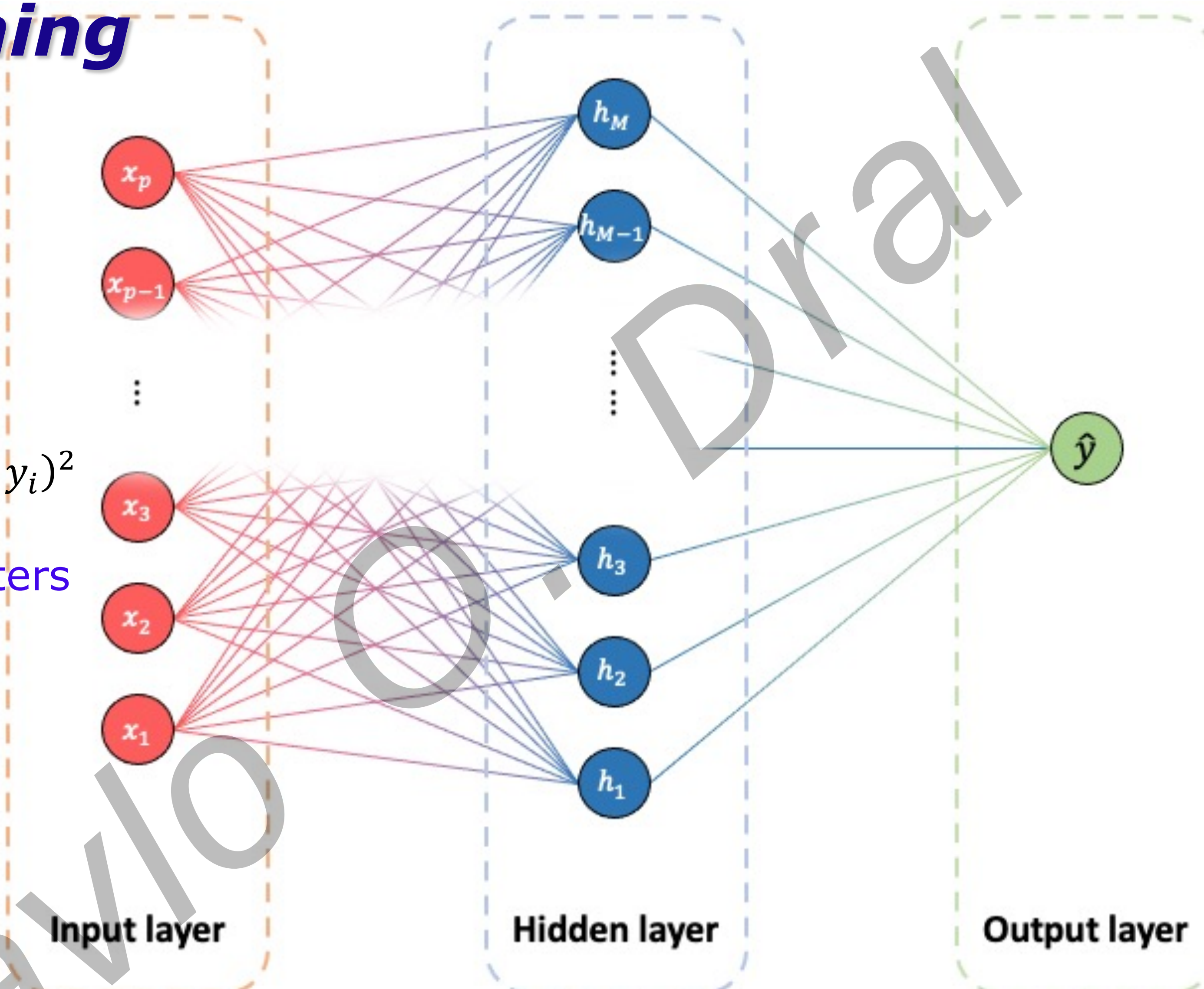
$$\arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N_{tr}} (f(\mathbf{x}_i;\boldsymbol{\theta}) - y_i)^2$$

$\boldsymbol{\theta} = \mathbf{w}, \boldsymbol{\alpha}, \mathbf{a}, b$ parameters

$\mathbf{w}, \boldsymbol{\alpha}$ … weights

$\mathbf{a}$, b… biases

$f(\mathbf{x}; \boldsymbol{\alpha}, \mathbf{a}, \mathbf{w}, b) =$
$f^{(2)}(\mathbf{h}; \mathbf{w}, b)$



P. O. Dral, A. Kananenka, F. Ge, B.-X. Xue, Neural Networks. In *Quantum Chemistry in the Age of Machine Learning*, 1st ed.; P. O. Dral, Ed. Elsevier: 2022. In press.

Training NNs

To train NN means to find its weights $\boldsymbol{\theta}$ usually by solving this minimization task:

$$\arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N_{\mathrm{tr}}} \left(f(\mathbf{x}_i;\boldsymbol{\theta}) - y_i\right)^2$$

To avoid overfitting this solution can be regularized using weight decay approach (recall ridge regression and KRR):

$$\arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N_{tr}} \left(f(\mathbf{x}_i;\boldsymbol{\theta}) - y_i\right)^2 + \lambda \sum_{j=1}^{N_p} \theta_j^2$$

$\boldsymbol{\theta} = \mathbf{w}, \boldsymbol{\alpha}, \mathbf{a}, b$ parameters

$$\hat{y} = f(\mathbf{x};\boldsymbol{\alpha},\mathbf{a},\mathbf{w},b) = b + w_1 h_1(\mathbf{x};\boldsymbol{\alpha}_1,a_1) + \cdots + w_M h_M(\mathbf{x};\boldsymbol{\alpha}_M,a_M) = \mathbf{h}^T\mathbf{w} + b$$

$$h_m(\mathbf{x};\boldsymbol{\alpha}_m,a_m) = g\left(a_m + \alpha_{m1}x_1 + \alpha_{m2}x_2 + \cdots + \alpha_{mp}x_p\right) = g(\mathbf{x}^T\boldsymbol{\alpha}_m + a_m)$$

P. O. Dral, A. Kananenka, F. Ge, B.-X. Xue, Neural Networks. In *Quantum Chemistry in the Age of Machine Learning*, 1st ed.; P. O. Dral, Ed. Elsevier: 2022. In press.

Training NNs

To train NN means to find its weights $\boldsymbol{\theta}$ usually by solving this minimization task:

$$\arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N_{\mathrm{tr}}} (f(\mathbf{x}_i; \boldsymbol{\theta}) - y_i)^2$$

To avoid overfitting this solution can be regularized using weight decay approach (recall ridge regression and KRR):

$$\arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N_{tr}} (f(\mathbf{x}_i; \boldsymbol{\theta}) - y_i)^2 + \lambda \sum_{j=1}^{N_p} \theta_j^2$$

$\boldsymbol{\theta} = \mathbf{w}, \boldsymbol{\alpha}, \mathbf{a}, b$ parameters

$$\hat{y} = f(\mathbf{x}; \boldsymbol{\alpha}, \mathbf{a}, \mathbf{w}, b) = b + w_1 h_1(\mathbf{x}; \boldsymbol{\alpha}_1, a_1) + \cdots + w_M h_M(\mathbf{x}; \boldsymbol{\alpha}_M, a_M) = \mathbf{h}^T \mathbf{w} + b$$

$$h_m(\mathbf{x}; \boldsymbol{\alpha}_m, a_m) = g\left(a_m + \alpha_{m1} x_1 + \alpha_{m2} x_2 + \cdots + \alpha_{mp} x_p\right) = g(\mathbf{x}^T \boldsymbol{\alpha}_m + a_m)$$

In contrast to linear regression and kernel methods, closed solution is unknown

P. O. Dral, A. Kananenka, F. Ge, B.-X. Xue, Neural Networks. In *Quantum Chemistry in the Age of Machine Learning*, 1st ed.; P. O. Dral, Ed. Elsevier: 2022. In press.

Training NNs: Issues

Issues with NNs:

In contrast to linear regression and kernel methods, no closed solution exists

Solutions are unstable and difficult to find.

Computationally expensive optimization problem should be solved and it therefore often can be speed up by using GPUs instead of CPUs.

GPUs are however much more expensive and difficult to get and optimization is still quite slow.

One of the popular approaches for fitting is back-propagation.

P. O. Dral, A. Kananenka, F. Ge, B.-X. Xue, Neural Networks. In *Quantum Chemistry in the Age of Machine Learning*, 1st ed.; P. O. Dral, Ed. Elsevier: 2022. In press.

Training NNs: Issues

Back-propagation:

$$L(\boldsymbol{\theta}) = \sum_{i=1}^{N} (f(\mathbf{x}_i; \boldsymbol{\theta}) - y_i)^2$$

gradient descent update with learning rate γ

$$\theta_k^{(r+1)} = \theta_k^{(r)} - \gamma \frac{\partial L(\boldsymbol{\theta})}{\partial \theta_k}$$

Well parallelized:

$$L(\boldsymbol{\theta}) = \sum_{i=1}^{N} L_i = \sum_{i=1}^{N} (f(\mathbf{x}_i; \boldsymbol{\theta}) - y_i)^2$$

The training set is often split into the minibatches (batches)

Update of parameters after the sweep over the entire training set is called an *epoch*.

P. O. Dral, A. Kananenka, F. Ge, B.-X. Xue, Neural Networks. In *Quantum Chemistry in the Age of Machine Learning*, 1st ed.; P. O. Dral, Ed. Elsevier: 2022. In press.

Training NNs: Issues

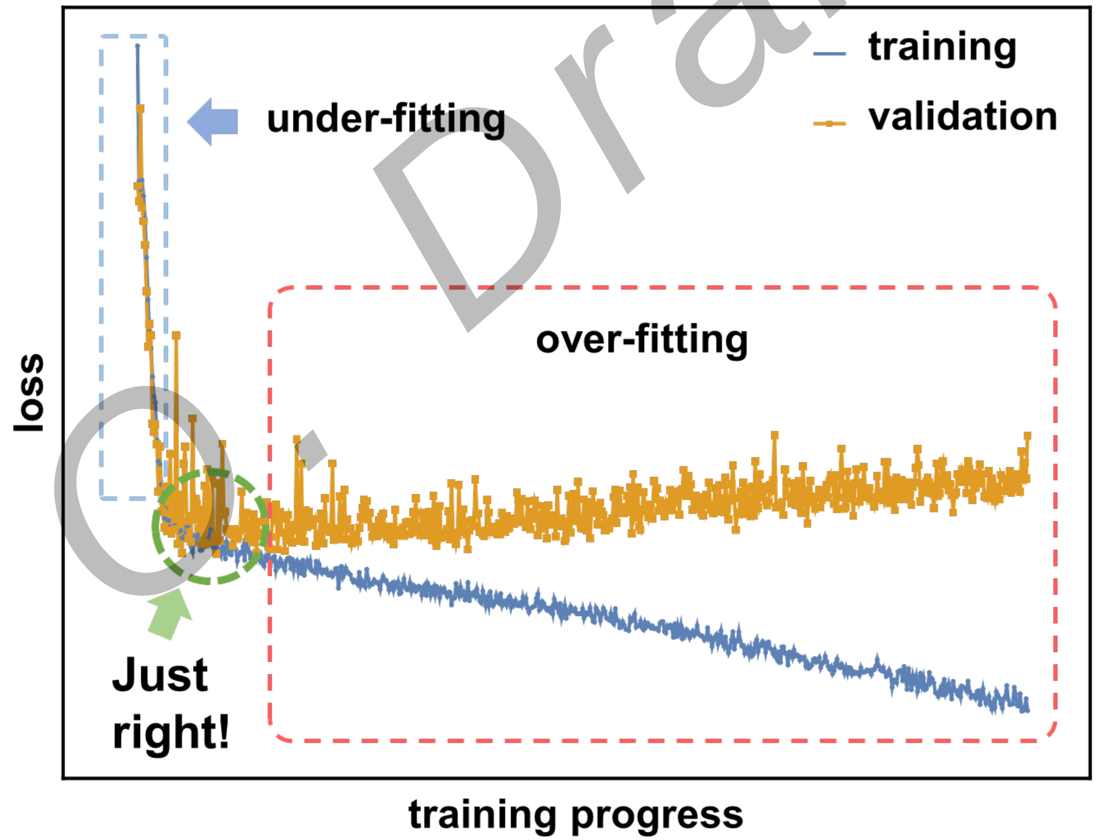
Issues with NNs:

Overfitting, regularization methods to deal with it:

- weight decay

$$\arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N_{tr}} (f(\mathbf{x}_i; \boldsymbol{\theta}) - y_i)^2 + \lambda \sum_{j=1}^{N_p} \theta_j^2$$

- early stopping
- data augmentation



P. O. Dral, A. Kananenka, F. Ge, B.-X. Xue, Neural Networks. In *Quantum Chemistry in the Age of Machine Learning*, 1st ed.; P. O. Dral, Ed. Elsevier: 2022. In press.

Training NNs: Issues

Issues with NNs:

- Input values should be scaled, usually standardized to center the inputs and scale them so that their standard deviation is 1 (Z-score normalization)
- It is also important to center reference data
- Number of hidden layers and units should be adjusted often by manual experimentation

P. O. Dral, A. Kananenka, F. Ge, B.-X. Xue, Neural Networks. In *Quantum Chemistry in the Age of Machine Learning*, 1st ed.; P. O. Dral, Ed. Elsevier: 2022. In press.

Training NNs: Issues

Issues with NNs:

- Initial guess of weights strongly influences the final parameter values
- Starting with zero values prevents back-propagation algorithm to find better solutions
- Starting with too large values often leads to large generalization errors

P. O. Dral, A. Kananenka, F. Ge, B.-X. Xue, Neural Networks. In *Quantum Chemistry in the Age of Machine Learning*, 1st ed.; P. O. Dral, Ed. Elsevier: 2022. In press.

Training NNs: Issues

Issues with NNs:

- Initial guess of weights strongly influences the final parameter values
- Starting with zero values prevents back-propagation algorithm to find better solutions
- Starting with too large values often leads to large generalization errors

Thus, one can get lot of different NNs fitted on the same data!

One can exploit this:
- Take average of multiple NNs to get more stable prediction
- Use deviation between NN predictions to estimate prediction uncertainty (e.g. useful in active learning)

P. O. Dral, A. Kananenka, F. Ge, B.-X. Xue, Neural Networks. In *Quantum Chemistry in the Age of Machine Learning*, 1st ed.; P. O. Dral, Ed. Elsevier: 2022. In press.

Active learning with several NNs

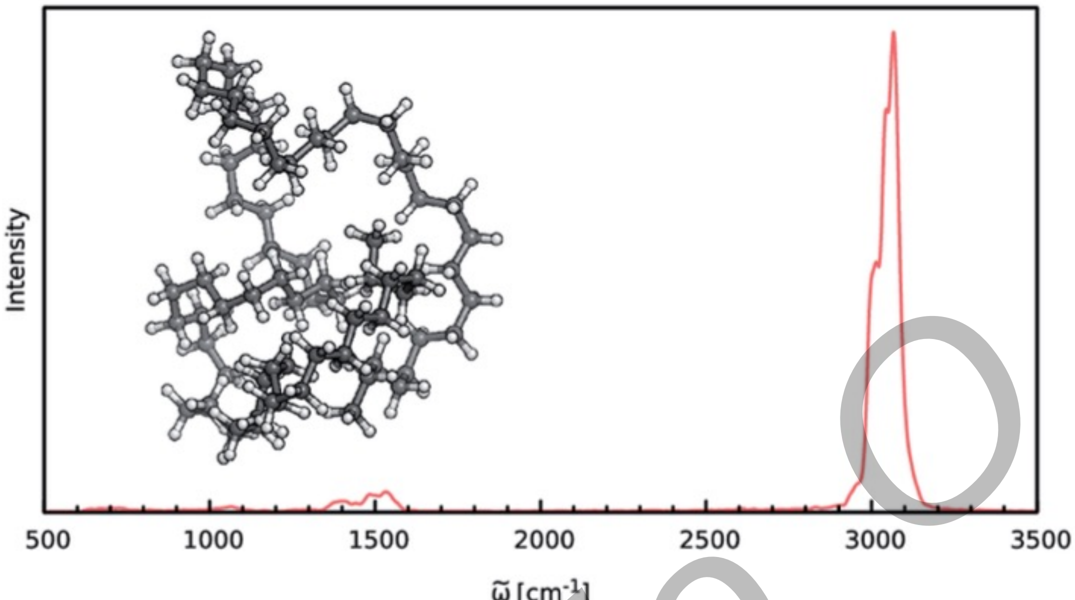
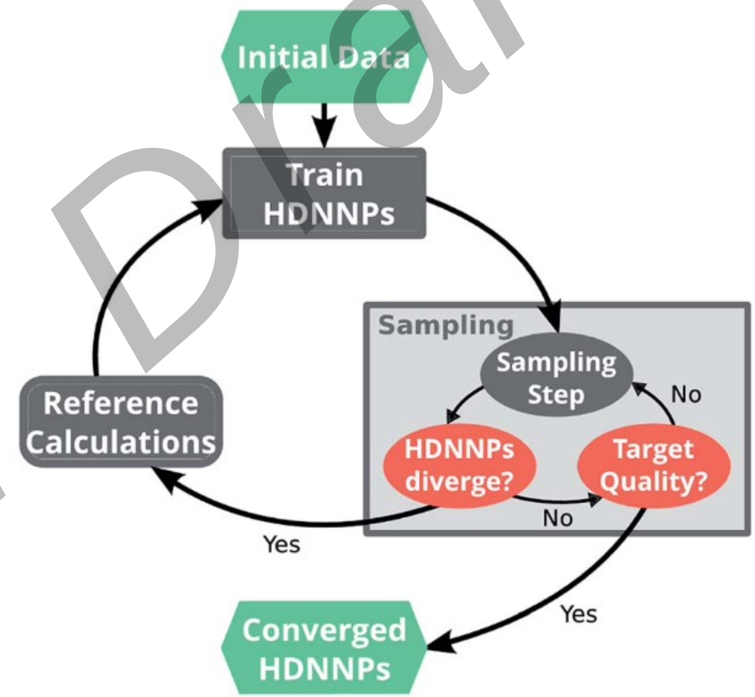


Fig. 6 IR spectrum of the $C_{69}H_{140}$ alkane as predicted by the ML model based on the B2PLYP method.



L5

M. Gastegger, J. Behler, P. Marquetand, *Chem. Sci.* **2017**, *8*, 6924

ML algorithms

Deep learning is based on neural networks (NN) with large depth (for feed-forward neural network – more than one hidden unit) in contrast to shallow neural network

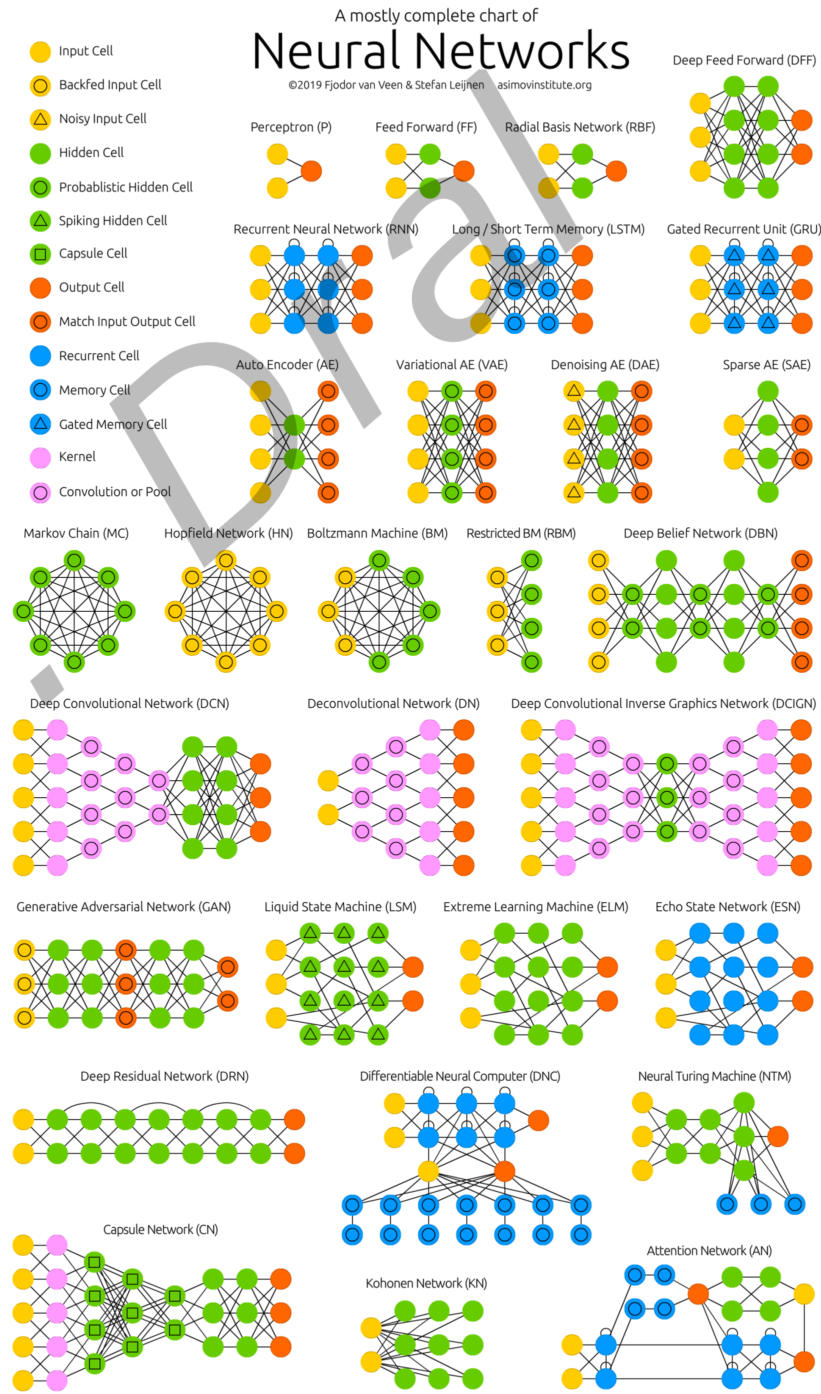
Some of other types of neural networks:

- Convolutional networks
- Recurrent neural networks
- Autoencoders

P. O. Dral, A. Kananenka, F. Ge, B.-X. Xue, Neural Networks. In *Quantum Chemistry in the Age of Machine Learning*, 1st ed.; P. O. Dral, Ed. Elsevier: 2022. In press.

The NN zoo

https://www.asimovinstitute.org/neural-network-zoo/



Parametric vs nonparametric algorithms

Comparison of KMs and NNs

What method to choose?
Kernel methods or neural networks?

Some of the ML Algorithms

f(x; **parameters**)

Linear regression

$$f(\mathbf{x}_i; \boldsymbol{\beta}) = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots$$

Number of parameters is fixed: parametric model

Neural networks are also parametric models

Kernel ridge regression (KRR)

$$f(\mathbf{x}_i; \mathbf{p}) = \sum_{j=1}^{N_{tr}} \alpha_j k\left(\mathbf{x}_i, \mathbf{x}_j; \mathbf{b}\right)$$

Number of parameters depends on number of training points: nonparametric model, e.g. KRR

Some of the ML Algorithms

All have some advantages and disadvantages, but often provide results with similar accuracy.

In many cases **it is not possible to claim that one fitting method is better than another**. The **choice will depend on experience and taste**.[1]

[1] Manzhos, Dawes, Carrington, *Int. J. Quantum Chem.* **2015**, *115*, 1012

Some of the ML Algorithms

However: You should be aware of the **law of the hammer** and do not try to use a hammer for every problem only because you already have a hammer.

Source: Wikipedia

Some of the ML Algorithms

Kernel ridge regression/ Gaussian Processes	Neural Networks
+ More accurate	– Less accurate
– Slow for large training sets	+ Fast for large training sets

"some of [… neural] networks became Gaussian processes in **the limit of infinite size**" [1]

[1] Rasmussen, Williams, *Gaussian Processes for Machine Learning*. The MIT Press: Boston, **2006**

P. O. Dral, *J. Phys. Chem. Lett.* **2020**, *11*, 2336

Some of the ML Algorithms

TABLE I. RMSE (in kcal/mol) per isomer on the provided training, validation, and test sets in the PIP, BPNN, and GAP short range interaction two-body (2B) and three-body (3B) energy fitting.

	2B			3B		
	Training	Validation	Test	Training	Validation	Test
PIP	0.0349	0.0449	0.0494	0.0262	0.0463	0.0465
BPNN	0.0493	0.0784	0.0792	0.0318	0.0658	0.0634
GAP	0.0176	0.0441	0.0539	0.0052	0.0514	0.0517

Nguyen, Szekely, Imbalzano, Behler, Csanyi, Ceriotti, Gotz, Paesani, *J. Chem. Phys.* **2018**, *148*, 241725

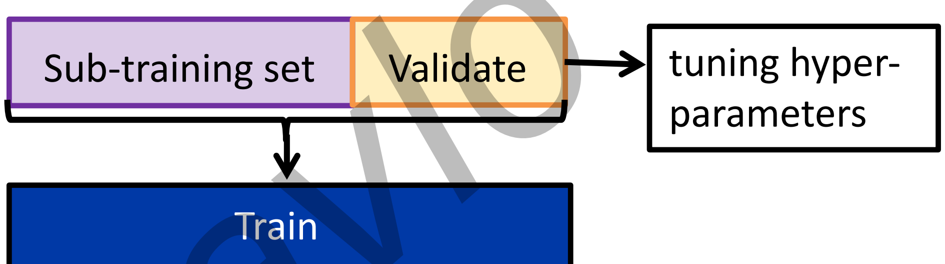
KRR-CM vs KREG

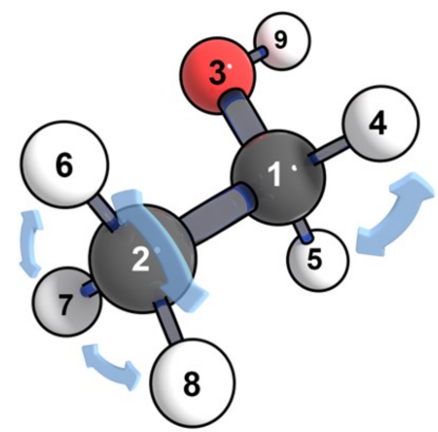
KRR-CM – kernel ridge regression with Gaussian kernel and Coulomb matrix descriptor

Test set RMSEs in kcal/mol

Number of training points	KRR-CM	KREG
100	**3.90±0.41**	**4.45±0.36**
2500	**0.70±0.02**	**0.52±0.01**

What do we mean by '100 training points?' Is validation set included?
Let's use the term **'sub-training set'**!

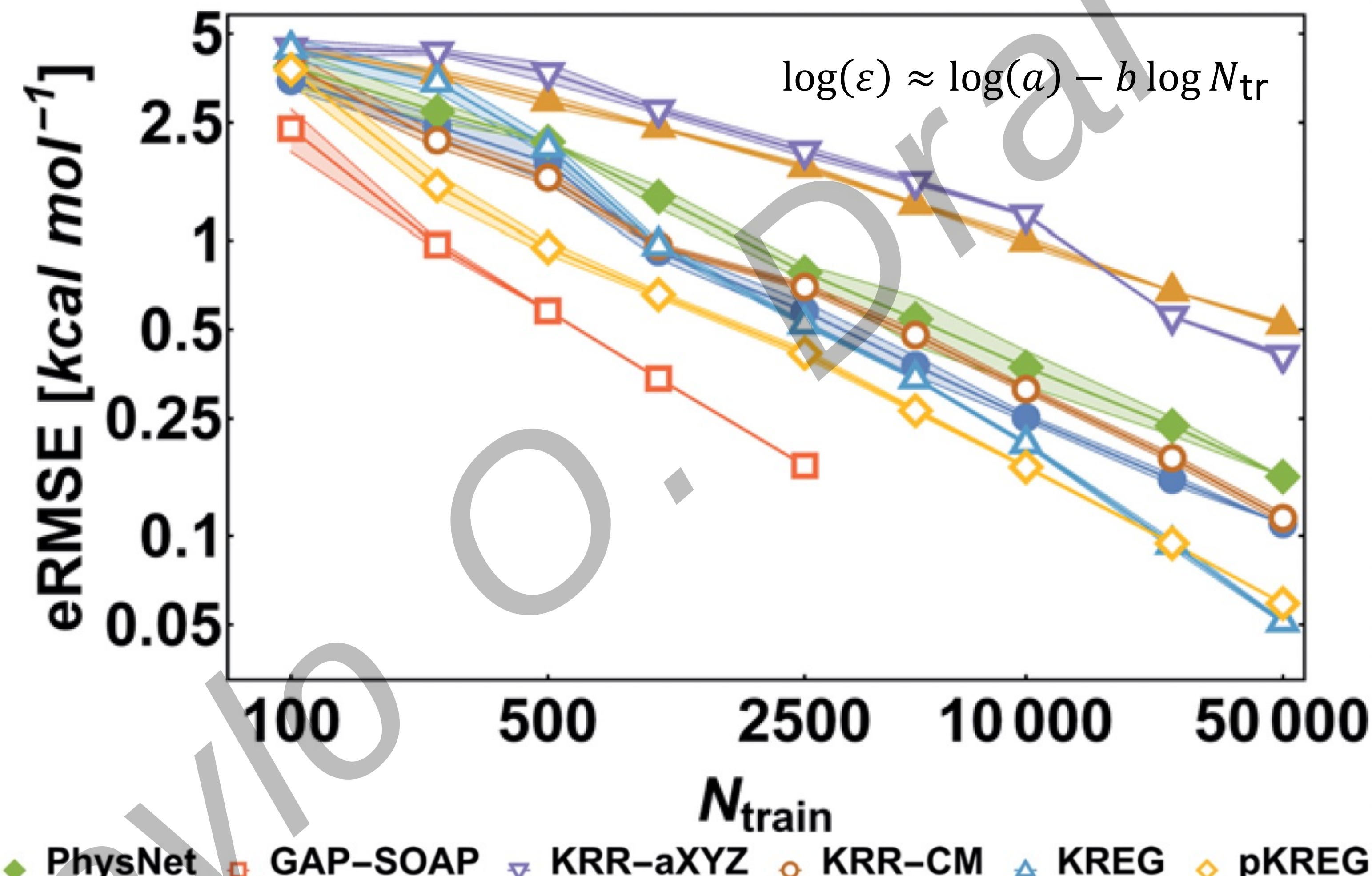




M. Pinheiro Jr, F. Ge, N. Ferré, P. O. Dral, M. Barbatti. *Chem. Sci.* **2021**, *12*, 14396–14413

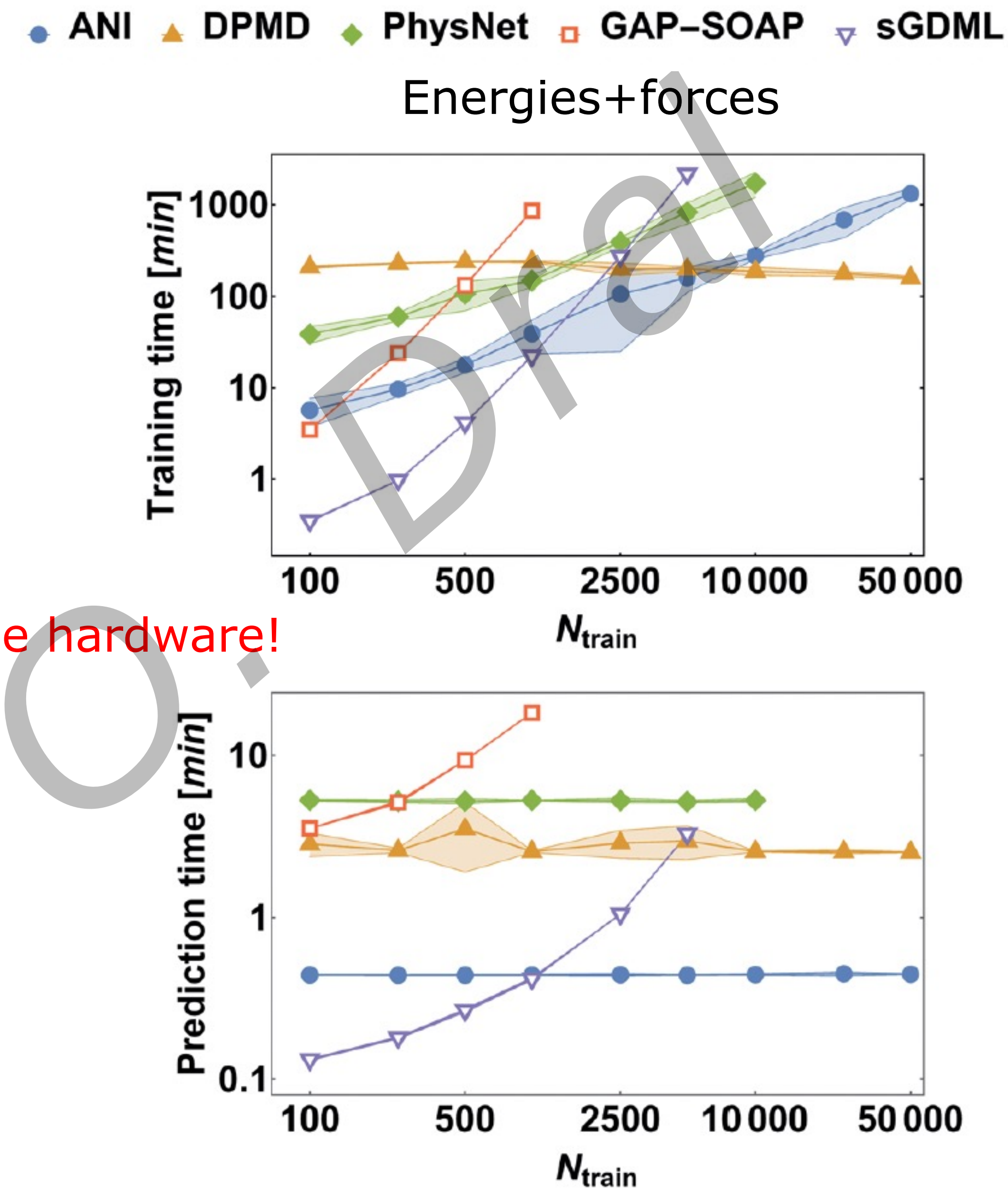
Learning curves (energies-only)

$$\varepsilon = \varepsilon_{\mathrm{a}} + \frac{a}{N_{\mathrm{tr}}^{b}}$$



M. Pinheiro Jr, F. Ge, N. Ferré, P. O. Dral, M. Barbatti. *Chem. Sci.* **2021**, *12*, 14396–14413
P. O. Dral, F. Ge, B.-X. Xue, Y.-F. Hou, M. Pinheiro Jr, J. Huang, M. Barbatti. *Top. Curr. Chem.* **2021,** *379*, 27

Timings!

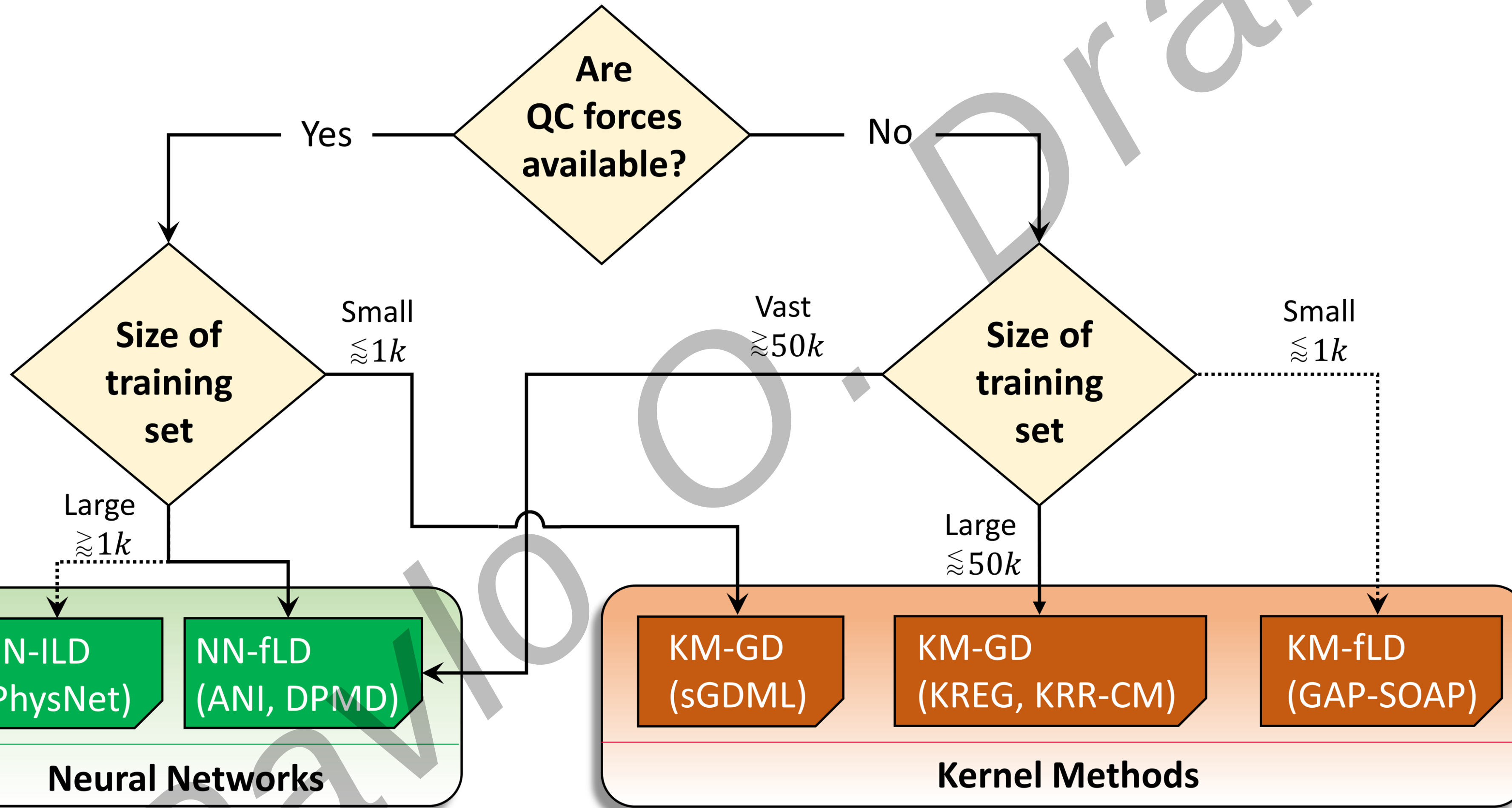


For small training sets kernel methods (open markers) are often both more accurate and faster for training and prediction than neural networks (filled markers)

The same hardware!

M. Pinheiro Jr, F. Ge, N. Ferré, P. O. Dral, M. Barbatti. *Chem. Sci.* **2021**, *12*, 14396–14413

Choosing the 'Right' ML Potential



M. Pinheiro Jr, F. Ge, N. Ferré, P. O. Dral, M. Barbatti. *Chem. Sci.* **2021**, *12*, 14396–14413

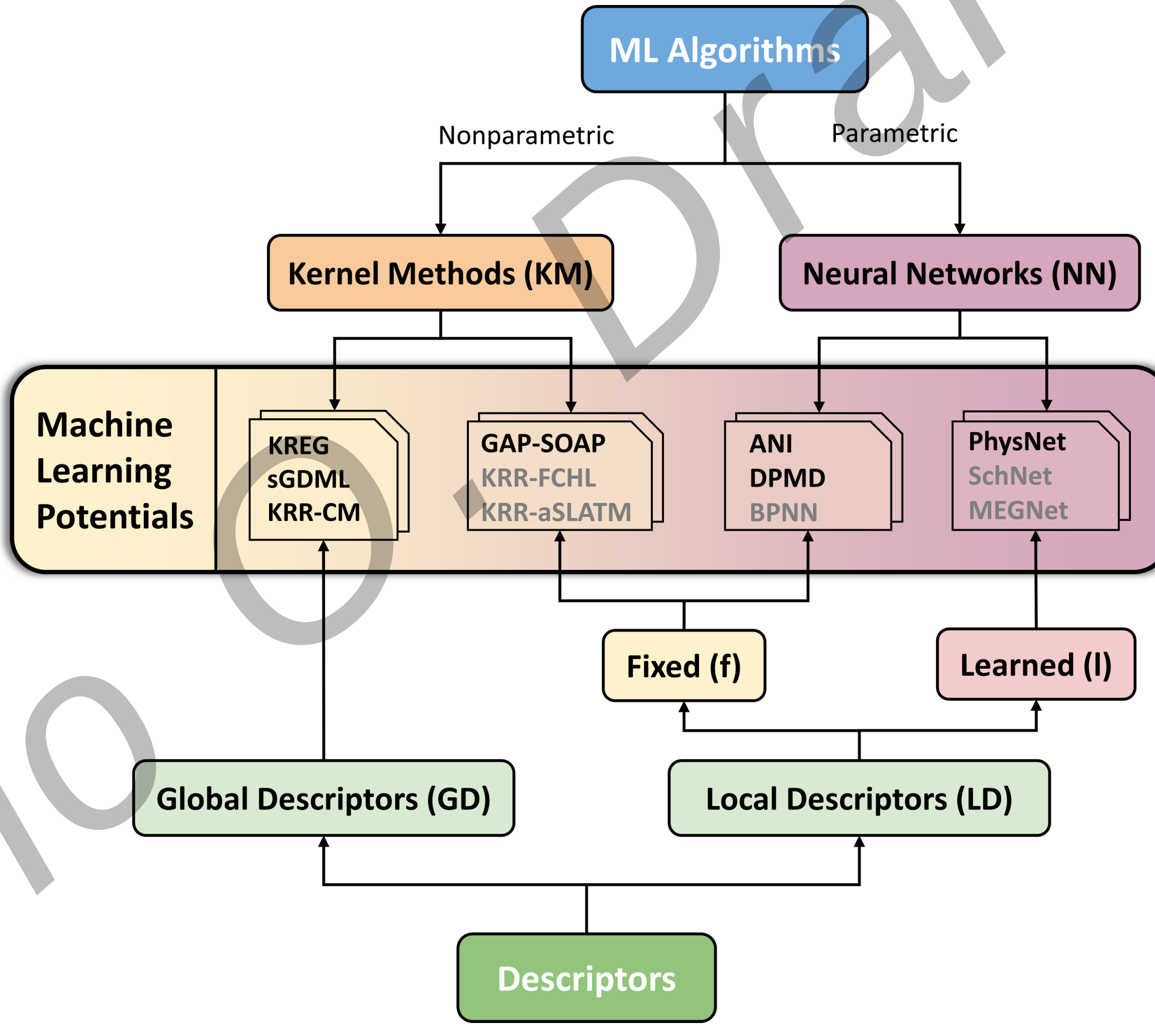
Zoo of machine learning potentials

Normal font – implemented models in MLatom

Greyed – not implemented in MLatom

MLatom.com

XACS



M. Pinheiro Jr, F. Ge, N. Ferré, P. O. Dral, M. Barbatti. *Chem. Sci.* **2021**, *12*, 14396–14413

Good ML method for quantum dynamics?

TABLE II. Mean absolute prediction errors (MAEs), training, and average single step prediction times of all KRR models used in this work. KRR-L, KRR-G, KRR-DP, KRR-E, and KRR-M ($n = 1, 2, 3, 4$) denote kernel ridge regression models with linear kernel, Gaussian kernel, decaying-periodic kernel, exponential kernel, and Matern kernel with $n = 1, 2, 3, 4$, respectively.

	Trainable	Mean absolute error		Time [s]	
Model	Parameters	Symmetric	Asymmetric	Training	Prediction
KRR-L	72,000	$1.2{\cdot}10^{-2}$	$6.5{\cdot}10^{-2}$	196	1.6
KRR-G	72,000	$4.7{\cdot}10^{-4}$	$1.2{\cdot}10^{-3}$	220	1.6
KRR-DP	72,000	$4.3{\cdot}10^{-4}$	$2.0{\cdot}10^{-3}$	257	1.4
KRR-E	72,000	$2.1{\cdot}10^{-3}$	$3.3{\cdot}10^{-3}$	222	1.6
KRR-M1	72,000	$2.4{\cdot}10^{-4}$	$1.3{\cdot}10^{-3}$	260	1.6
KRR-M2	72,000	$2.2{\cdot}10^{-4}$	$2.7{\cdot}10^{-3}$	259	1.7
KRR-M3	72,000	$2.0{\cdot}10^{-4}$	$2.3){\cdot}10^{-3}$	279	1.7
KRR-M4	72,000	$2.3{\cdot}10^{-4}$	$2.1{\cdot}10^{-3}$	273	1.7

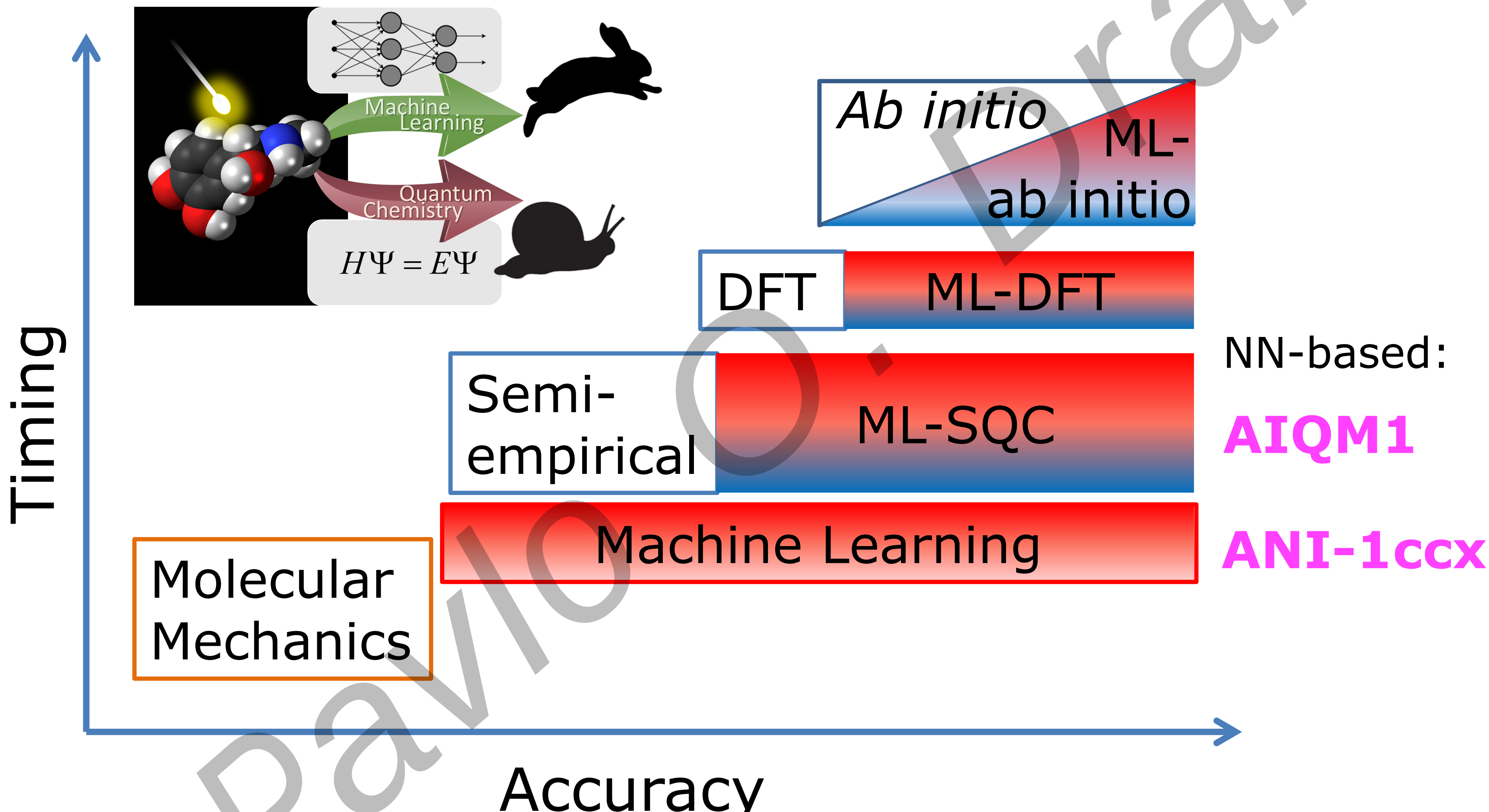
TABLE I. Hyperparameters, mean absolute prediction errors, total number of trainable parameters, training, and average single step prediction times of all ANN models studied in this work. For each recurrent layer the number of units is shown. The number of kernels X and kernel sizes Y for each convolutional layer is shown in parenthesis as (X,Y).

	Trainable	Layers		Mean absolute error		Time [s]	
Model	Parameters	Layer 1	Layer 2	Symmetric	Asymmetric	Training	Prediction
1D CNN	530,258	(235,16)	(125,7)	$1.55{\cdot}10^{-3}$	$4.84{\cdot}10^{-2}$	465	3.8
FFNN	520,045	754	646	$1.32{\cdot}10^{-3}$	$3.70{\cdot}10^{-2}$	82	3.3
Recurrent Neural Networks							
LSTM	528,577	15	49	$1.58{\cdot}10^{-3}$	$2.35{\cdot}10^{-2}$	623	6.1
GRU	553,453	60	50	$2.05{\cdot}10^{-3}$	$2.57{\cdot}10^{-2}$	668	4.4
RNN	535,468	65	50	$3.00{\cdot}10^{-3}$	$6.17{\cdot}10^{-2}$	302	4.2
Convolutional Recurrent Neural Networks							
CLSTM	501,965	(28,16)	71	$1.17{\cdot}10^{-3}$	$2.50{\cdot}10^{-2}$	279	4.1
CGRU	515,806	(55,16)	73	$1.38{\cdot}10^{-3}$	$2.14{\cdot}10^{-2}$	294	4.7
CRNN	513,673	(243,16)	73	$1.46{\cdot}10^{-3}$	$3.61{\cdot}10^{-2}$	197	5.4
Convolutional Bidirectional Recurrent Neural Networks							
CBLSTM	568,022	(109,16)	39	$1.17{\cdot}10^{-3}$	$2.84{\cdot}10^{-2}$	333	3.8
CBGRU	514,860	(55,16)	37	$1.54{\cdot}10^{-3}$	$3.68{\cdot}10^{-2}$	325	5.3
CBRNN	508,842	(297,16)	36	$2.50{\cdot}10^{-3}$	$3.58{\cdot}10^{-2}$	256	3.9
Bidirectional Recurrent Neural Networks							
BLSTM	511,809	6	24	$2.12{\cdot}10^{-3}$	$2.56{\cdot}10^{-2}$	635	5.2
BGRU	534,991	14	25	$2.28{\cdot}10^{-3}$	$2.48{\cdot}10^{-2}$	1109	6.7
BRNN	511,959	37	24	$6.95{\cdot}10^{-3}$	$4.27{\cdot}10^{-1}$	396	5.4

L. E. H. Rodrıguez, A. Ullah, K. J. R. Espinosa, P. O. Dral, A. A. Kananenka. A comparative study of different machine learning methods for dissipative quantum dynamics. **2022**, *submitted*. *arXiv*: https://arxiv.org/abs/2207.02417.

Machine learning

P. O. Dral, M. Barbatti, *Nat. Rev. Chem.* **2021**, *5*, 388



P. O. Dral, *J. Phys. Chem. Lett.* **2020**, *11*, 2336

Multi-fidelity data: Δ-learning

R. Ramakrishnan, P. O. Dral, M. Rupp, O. A. von Lilienfeld,
J. Chem. Theory Comput. **2015**, *11*, 2087

H_2 dissociation curve

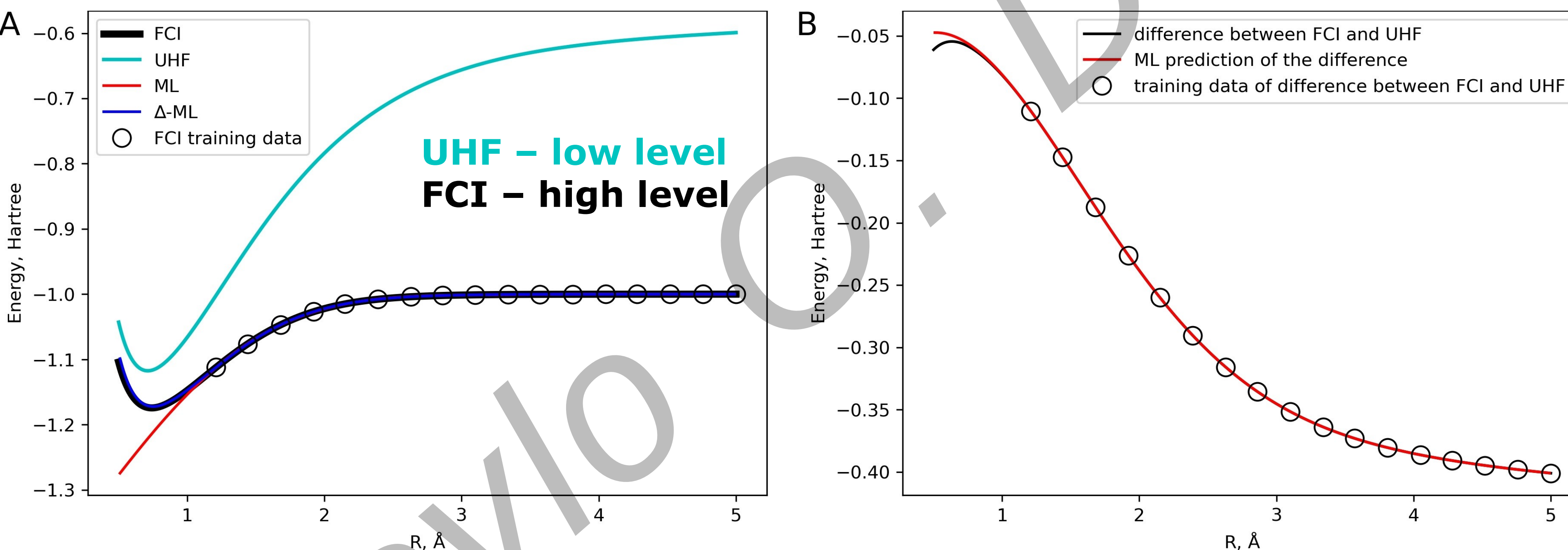


Figure: Pavlo O. Dral, Tetiana Zubatiuk, Bao-Xin Xue, Learning from multiple quantum chemical methods: Δ-learning, transfer learning, co-kriging, and beyond. In *Quantum Chemistry in the Age of Machine Learning*, Pavlo O. Dral, Ed. Elsevier: 2022, in press. Paperback ISBN: 9780323900492

Hierarchical ML (hML)
Combining Δ-ML with Sampling

$$\hat{y}_{\Sigma,i} = \sum_{M} \hat{y}_{M,i}(N_{\mathrm{tr},M})$$

Automatic procedure to find optimal training points for each delta-model

$$\frac{e_{\Sigma}}{e_{\Sigma,\mathrm{th}}} + \frac{t_{\Sigma}/t_{\Sigma,\max}}{1 - s_{\mathrm{th}}} + \frac{p}{e_{\Sigma,\mathrm{th}}}$$

$$\frac{\hat{y}_{M,i}(N_{\mathrm{tr},M})}{\hat{y}_{1,i}(N_{\mathrm{tr},M})} \approx r_{M/1,i} = \frac{y_{M,i}}{y_{1,i}}$$

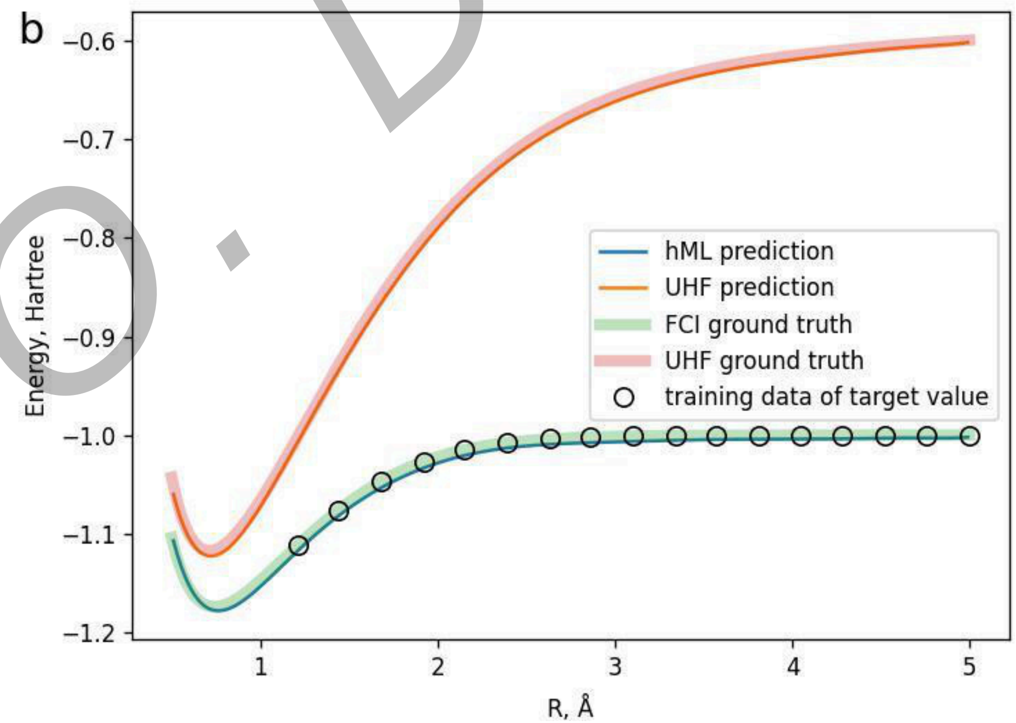
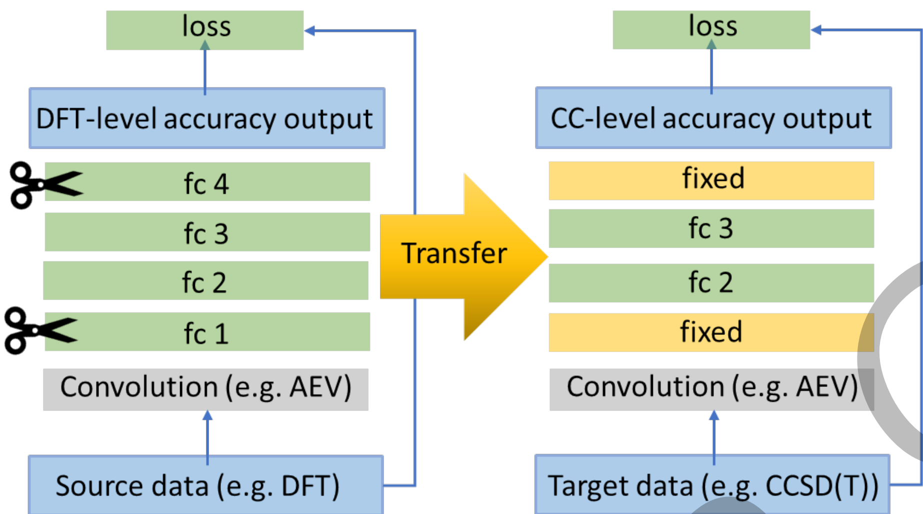
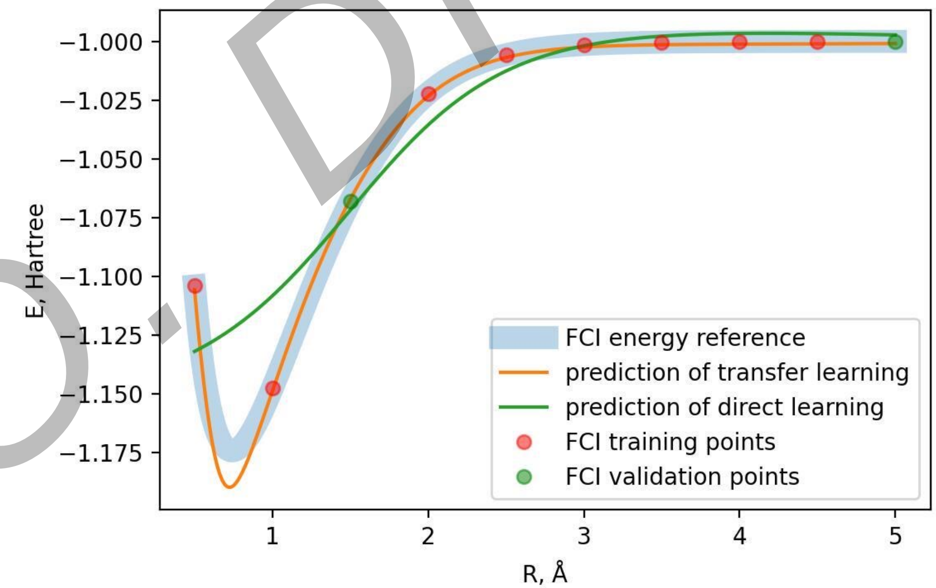


Figure: Pavlo O. Dral, Tetiana Zubatiuk, Bao-Xin Xue, Learning from multiple quantum chemical methods: Δ-learning, transfer learning, co-kriging, and beyond. In *Quantum Chemistry in the Age of Machine Learning*, Pavlo O. Dral, Ed. Elsevier: 2022, in press. Paperback ISBN: 9780323900492

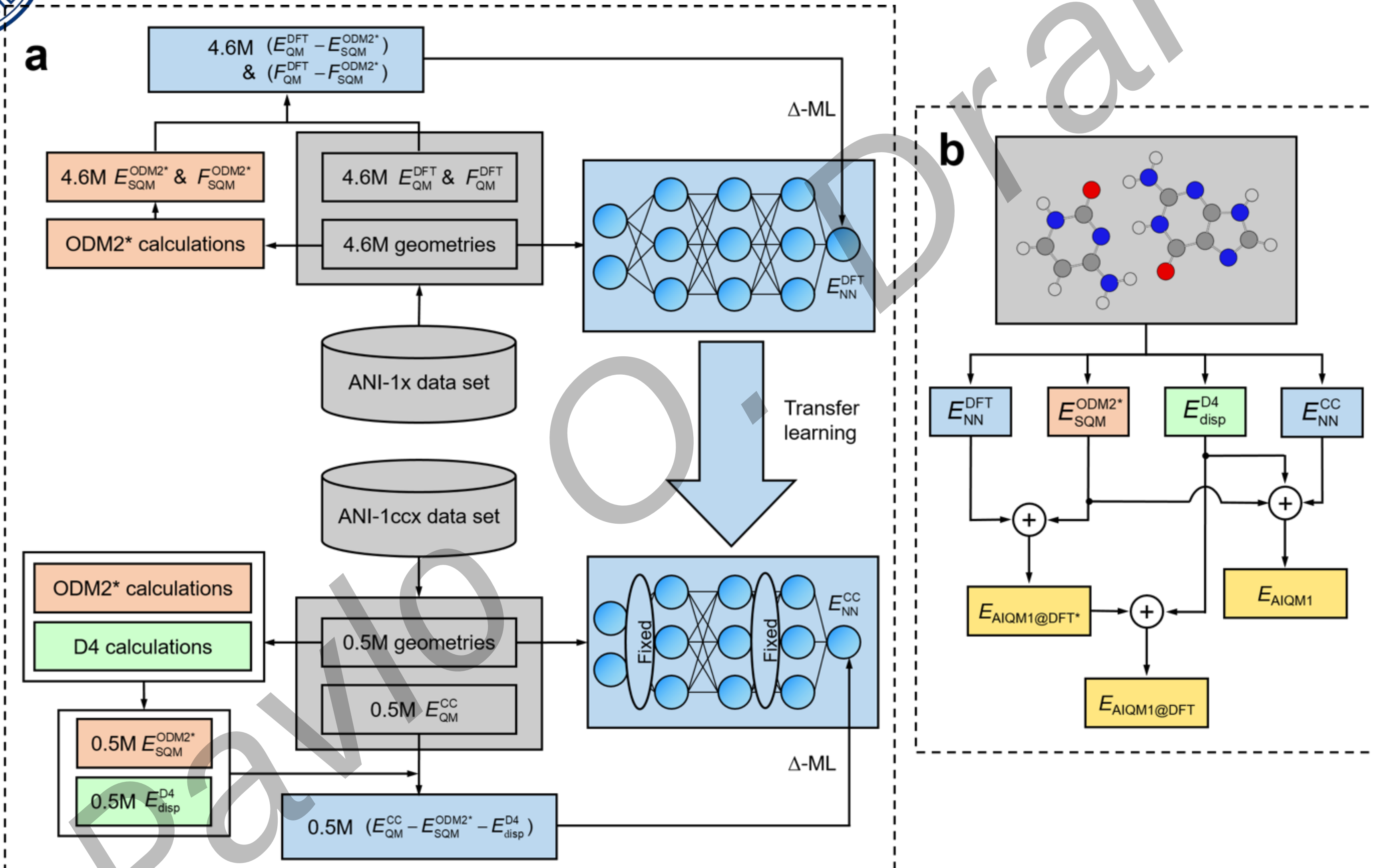
Multi-fidelity data: Transfer learning





Figures: Pavlo O. Dral, Tetiana Zubatiuk, Bao-Xin Xue, Learning from multiple quantum chemical methods: Δ-learning, transfer learning, co-kriging, and beyond. In *Quantum Chemistry in the Age of Machine Learning*, Pavlo O. Dral, Ed. Elsevier: 2022, in press. Paperback ISBN: 9780323900492

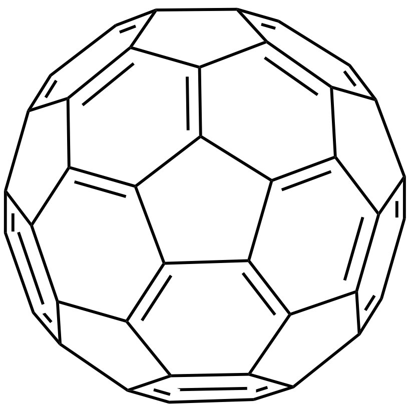
Training



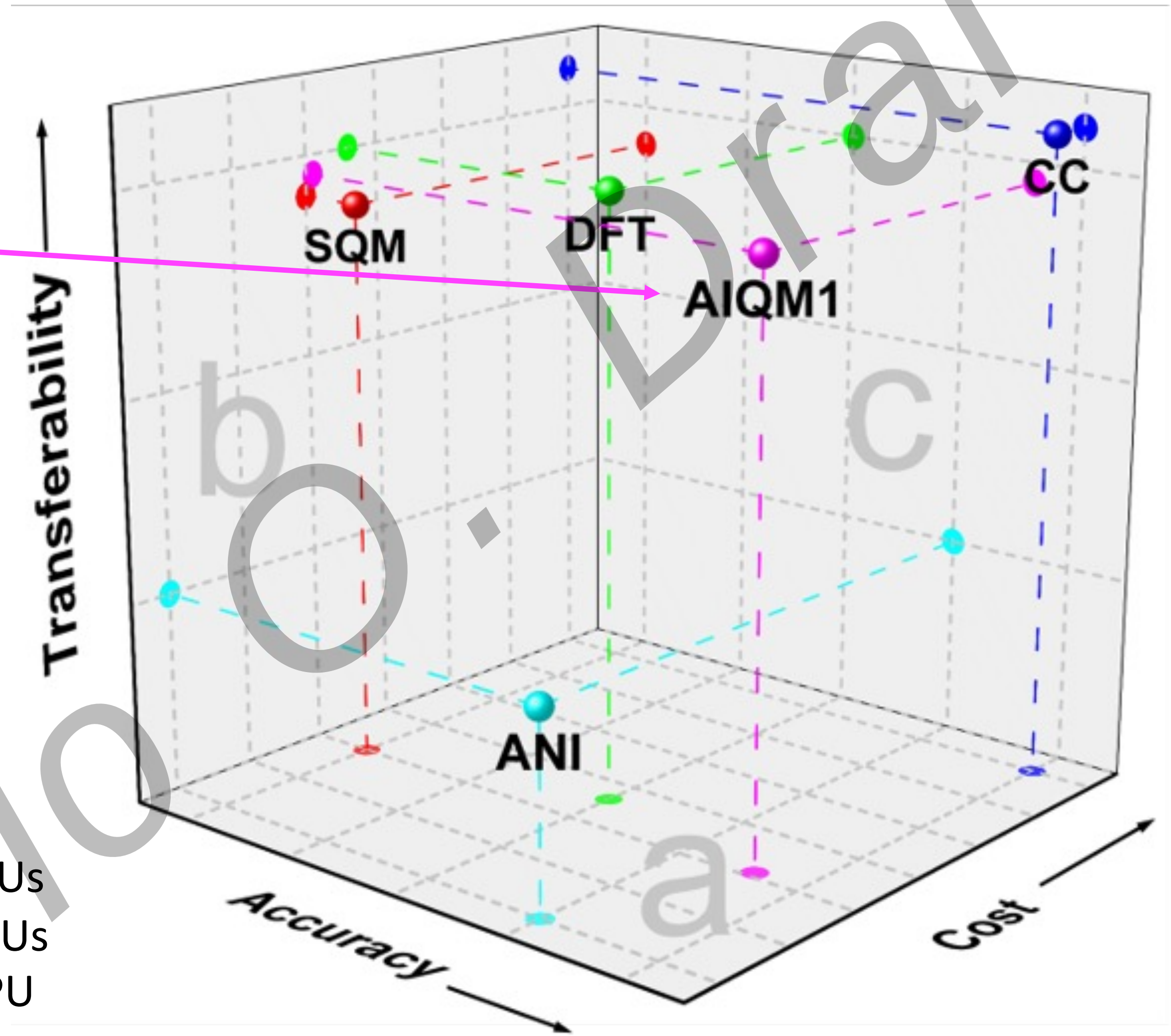
P. Zheng, R. Zubatyuk, W. Wu, O. Isayev, P. O. Dral, *Nat. Commun.* **2021**, *12*, 7022

Accuracy vs transferability vs cost

Artificial Intelligence-Enhanced Quantum Chemical Method 1
AIQM1



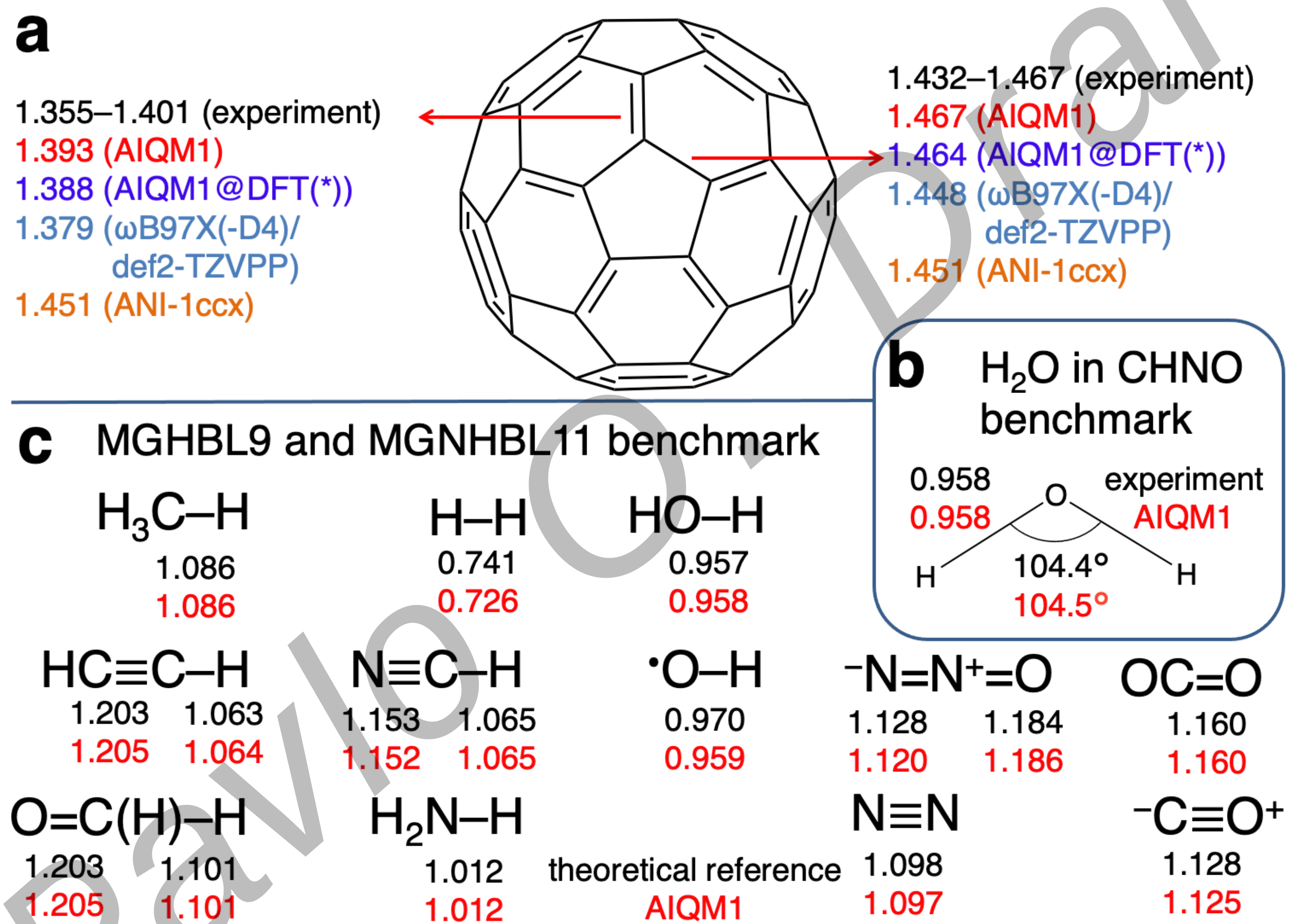
C_{60}



CCSD(T)*/CBS SP: 69 h /15 CPUs
DFT opt: 30 min /32 CPUs
AIQM1 opt: 14 sec / 1 CPU

P. Zheng, R. Zubatyuk, W. Wu, O. Isayev, P. O. Dral. Artificial Intelligence-Enhanced Quantum Chemical Method with Broad Applicability. *Nat. Commun.* **2021**, *12*, 7022.

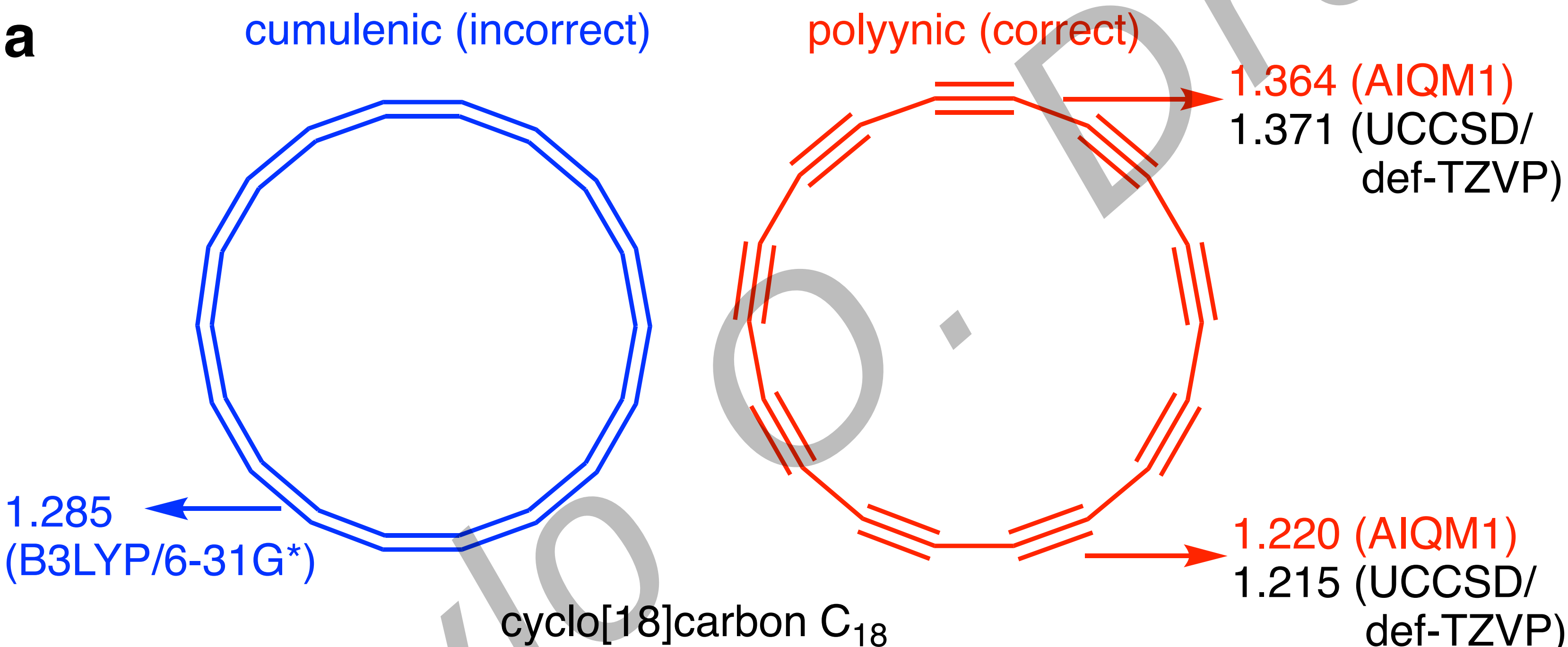
Ground-state geometries



P. Zheng, R. Zubatyuk, W. Wu, O. Isayev, P. O. Dral, *Nat. Commun.* **2021**, *12*, 7022

AIQM1 is better than many popular DFT (density functional theory) methods

Ground-state geometries

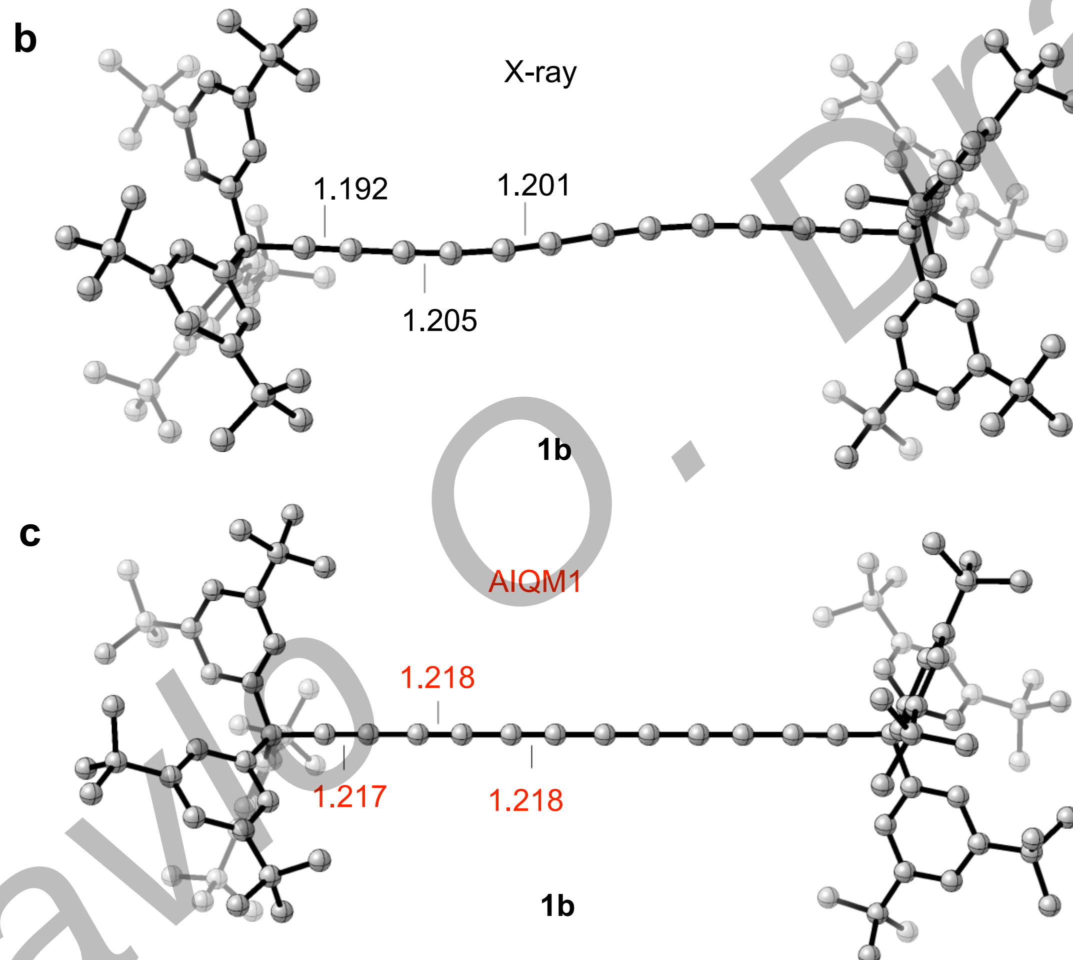


P. Zheng, R. Zubatyuk, W. Wu, O. Isayev, P. O. Dral, *Nat. Commun.* **2021**, *12*, 7022

… and sometimes better than

X-ray structure determination!

Ground-state geometries



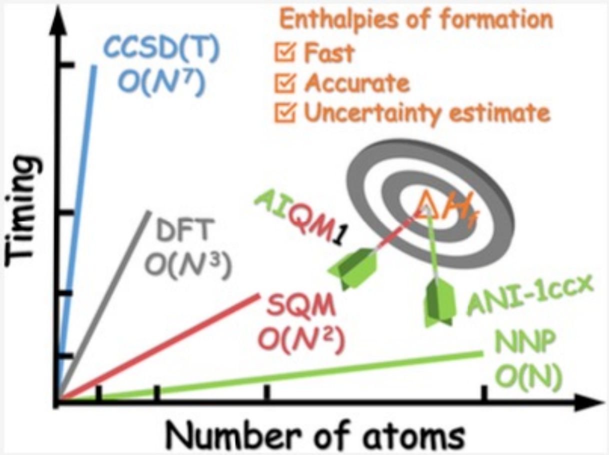
P. Zheng, R. Zubatyuk, W. Wu, O. Isayev, P. O. Dral, *Nat. Commun.* **2021**, *12*, 7022

Toward Chemical Accuracy in Predicting Enthalpies of Formation with General-Purpose Data-Driven Methods

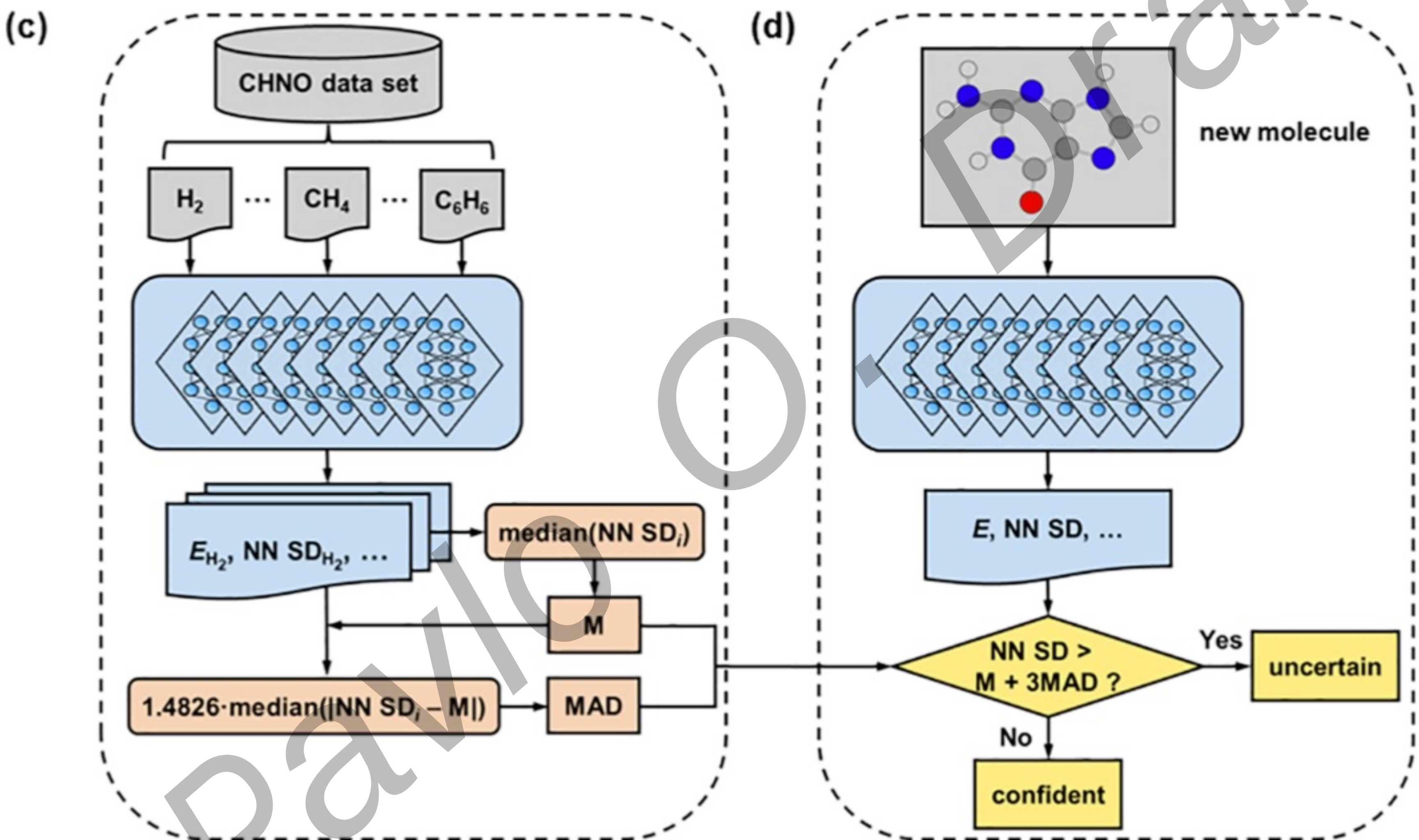
Peikun Zheng, Wudi Yang, Wei Wu, Olexandr Isayev,* and Pavlo O. Dral*

Cite This: *J. Phys. Chem. Lett.* 2022, 13, 3479–3491

ABSTRACT: Enthalpies of formation and reaction are important thermodynamic properties that have a crucial impact on the outcome of chemical transformations. Here we implement the calculation of enthalpies of formation with a general-purpose ANI-1ccx neural network atomistic potential. We demonstrate on a wide range of benchmark sets that both ANI-1ccx and our other general-purpose data-driven method AIQM1 approach the coveted chemical accuracy of 1 kcal/mol with the speed of semiempirical quantum mechanical methods (AIQM1) or faster (ANI-1ccx). It is remarkably achieved without specifically training the machine learning parts of ANI-1ccx or AIQM1 on formation enthalpies. Importantly, we show that these data-driven methods provide statistical means for uncertainty quantification of their predictions, which we use to detect and eliminate outliers and revise reference experimental data. Uncertainty quantification may also help in the systematic improvement of such data-driven methods.



Uncertainty quantification



P. Zheng, W. Yang, W. Wu, O. Isayev, P. O. Dral, *J. Phys. Chem. Lett.* **2022**, *13*, 3479

Heats of Formation: Outliers from Uncertainty Quantification

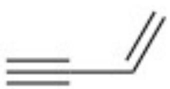
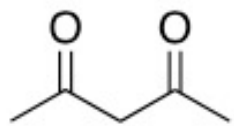
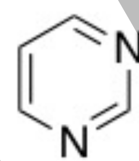
	ODM2	B3LYP/6-31G*	ωB97X/6-31G*	ωB97X-D-/6-31G*	ωB97X/def2-TZVPP	ωB97X-D4/def2-TZVPP	G4MP2	G4	AIQM1	AIQM1 (without outliers)	ANI-1ccx	ANI-1ccx (without outliers)
CHNO (ΔH_f)	2.63	6.72	4.09	3.83	3.20	2.74	0.90	0.75	0.84	0.60	1.76	0.92
BIGMOL20 (ΔH_f)	3.97	18.92	4.40	4.37	8.01	6.01	2.40	2.16	2.30	1.96	2.34	2.07
CONFORMERS30 (ΔH_f)	2.21	10.37	3.46	3.01	4.50	3.10	0.79	0.64	0.46	0.44	0.96	0.96
ISOMERS44 (ΔH_f)	1.16	8.08	3.57	3.53	4.52	3.78	0.44	0.37	0.42	0.42	1.34	0.59
ALKANES28 (ΔH_f)	1.15	11.17	7.07	6.26	5.12	3.06	0.59	0.53	0.97	0.24	2.67	0.65
G2 (ΔH_f)	2.59	4.99	4.31	4.07	2.60	2.39	0.60	0.52	0.97	0.44	2.62	0.62
G3 (ΔH_f)	2.90	8.66	3.48	3.26	4.28	3.30	0.74	0.68	0.73	0.68	1.08	0.97
HEDM-45 (ΔH_f)	5.55	9.06	4.33	3.90	3.46	3.48	1.20	2.27	3.80	0.96	2.87	1.25
PAH-103 (ΔH_f)	2.10	14.72	2.93	2.89	7.63	5.73	1.96	1.13	1.19	0.79	1.75	1.15
CHNO (ΔH_r)	1.92	1.75	1.41	1.39	1.32	1.29	0.94	1.02	1.06	0.22	2.63	0.68
CONFORMERS30 (ΔH_r)	1.36	1.19	1.09	1.09	1.29	1.28	1.28	1.27	1.35	1.20	1.37	1.37
ISOMERS44 (ΔH_r)	0.70	2.29	1.45	1.31	1.19	1.10	0.40	0.44	0.50	0.50	1.62	0.61
ALKANES28 (ΔH_r)	0.34	1.90	1.10	0.94	1.74	1.14	0.28	0.32	0.59	0.39	0.20	0.08
AF6 (ΔH_r)	1.32	3.07	0.55	1.68	1.28	0.73	1.02	1.05	0.45	—	3.67	—

Mean absolute errors (MAEs) in kcal/mol of benchmarked methods for various data sets. MAEs of AIQM1 and ANI-1ccx

P. Zheng, W. Yang, W. Wu, O. Isayev, P. O. Dral, *J. Phys. Chem. Lett.* **2022**, *13*, 3479

Revising Experimental Heats of Formation

CHNO data set

			
vinylacetylene	acetylacetone	1,3-pyrimidine	acrolein
original: 73	original: –91	exper.: 46.8	exper.: –18
revised: 70.4	revised: –89.9		
AIQM1: 69.1	AIQM1: –87.3	AIQM1: 44.5	AIQM1: –16.0
G4: 69.1	G4: –87.6	G4: 44.3	G4: –15.9

P. Zheng, W. Yang, W. Wu, O. Isayev, P. O. Dral, *J. Phys. Chem. Lett.* **2022**, *13*, 3479

MLatom

A Package for Atomistic Simulations with Machine Learning

Developed for practical and efficient application of machine learning in computational chemistry.

MLatom.com

XACS

Run calculations online or download MLatom

XACS

Xiamen Atomistic Computing Suite

XMVB

A Valence Bond Theory Based Quantum Chemistry Program

XMVB

XEDA

A General and Multipurpose Energy Decomposition Analysis Program

XEDA

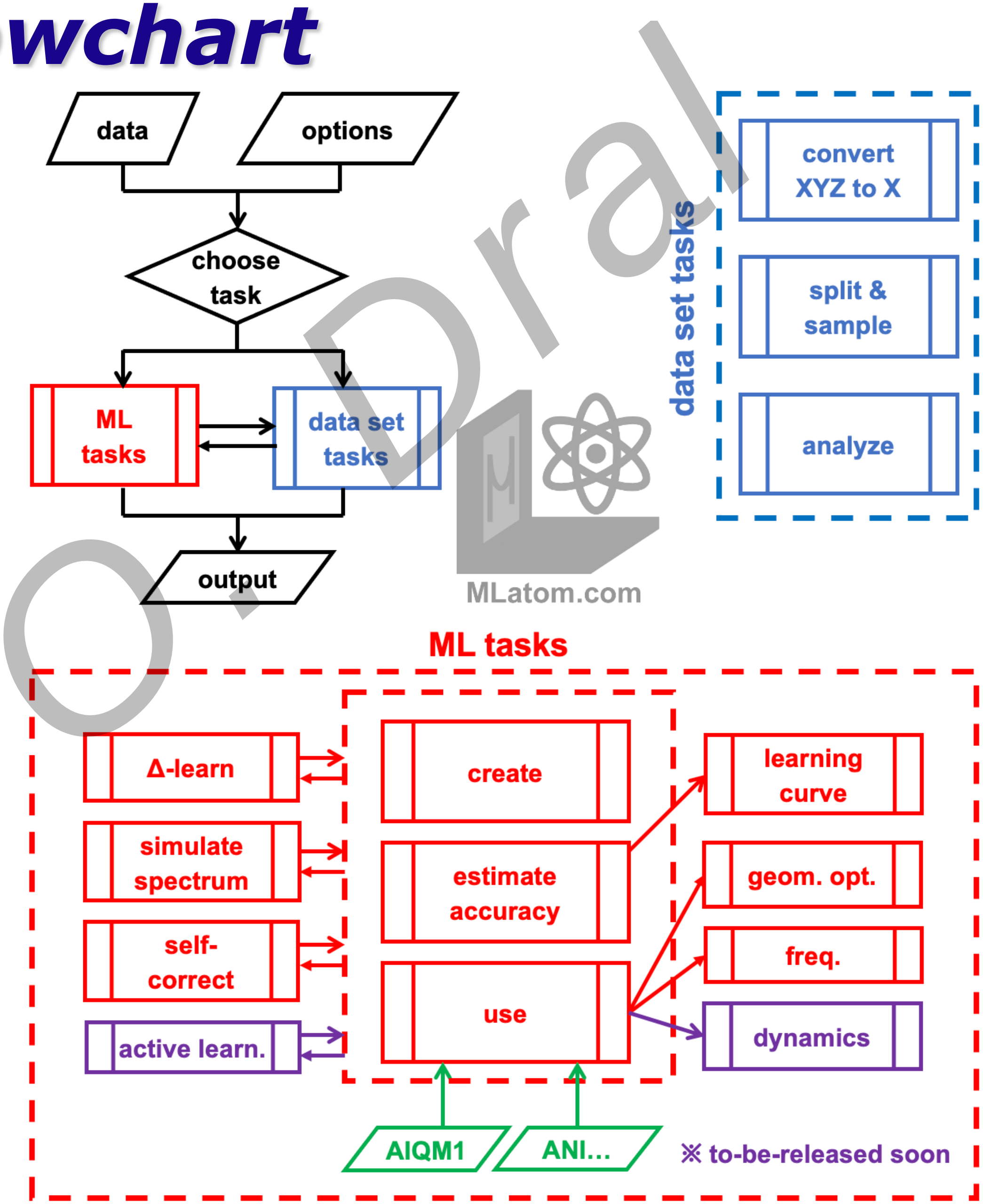
MLatom

A Package for Atomistic Simulations with Machine Learning

MLatom

Flowchart

P. O. Dral, F. Ge, B.-X. Xue, Y.-F. Hou,
M. Pinheiro Jr, J. Huang, M. Barbatti, **2021**, *379*, 27



3rd-party Interfaces

DeePMD-kit	deepmodeling.com
GAP	libatoms.github.io/GAP
sGDML	sgdml.org
TorchANI	aiqm.github.io/torchani
PhysNet	github.com/MMunibas/PhysNet
ASE	wiki.fysik.dtu.dk/ase
Hyperopt	hyperopt.github.io/hyperopt
Sparrow	scine.ethz.ch/download/sparrow
NX	newtonx.org
Gaussian	gaussian.com
MNDO	mndo.kofo.mpg.de

Acknowledgements

Thank you for your attention!

Funding:

PCOSS

国家自然科学基金委员会
National Natural Science Foundation of China