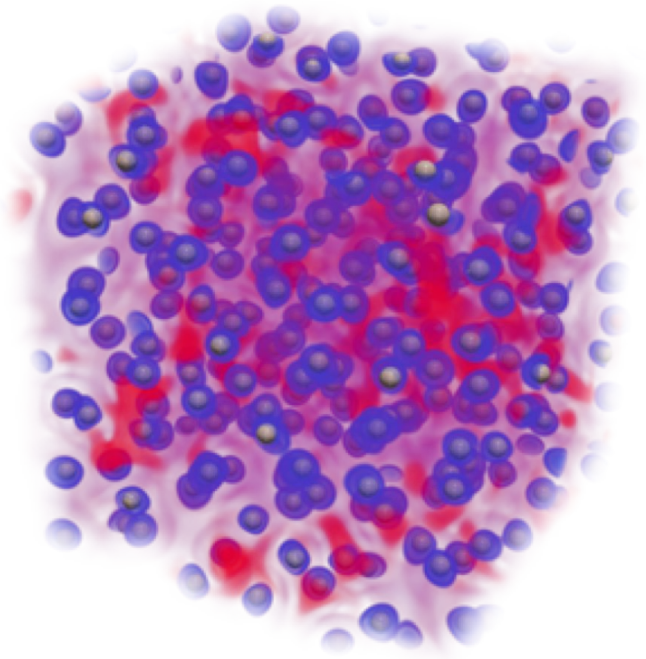


Atomistic Simulation Methods

Machine Learning Interatomic Potentials (2/2)

Helmholtz-Zentrum Dresden-Rossendorf · Center for Advanced Systems Understanding · Machine Learning for Materials Design · Attila Cangi · a.cangi@hzdr.de

INSTITUTE OF

PARTICIPATING INSTITUTIONS

FUNDED BY

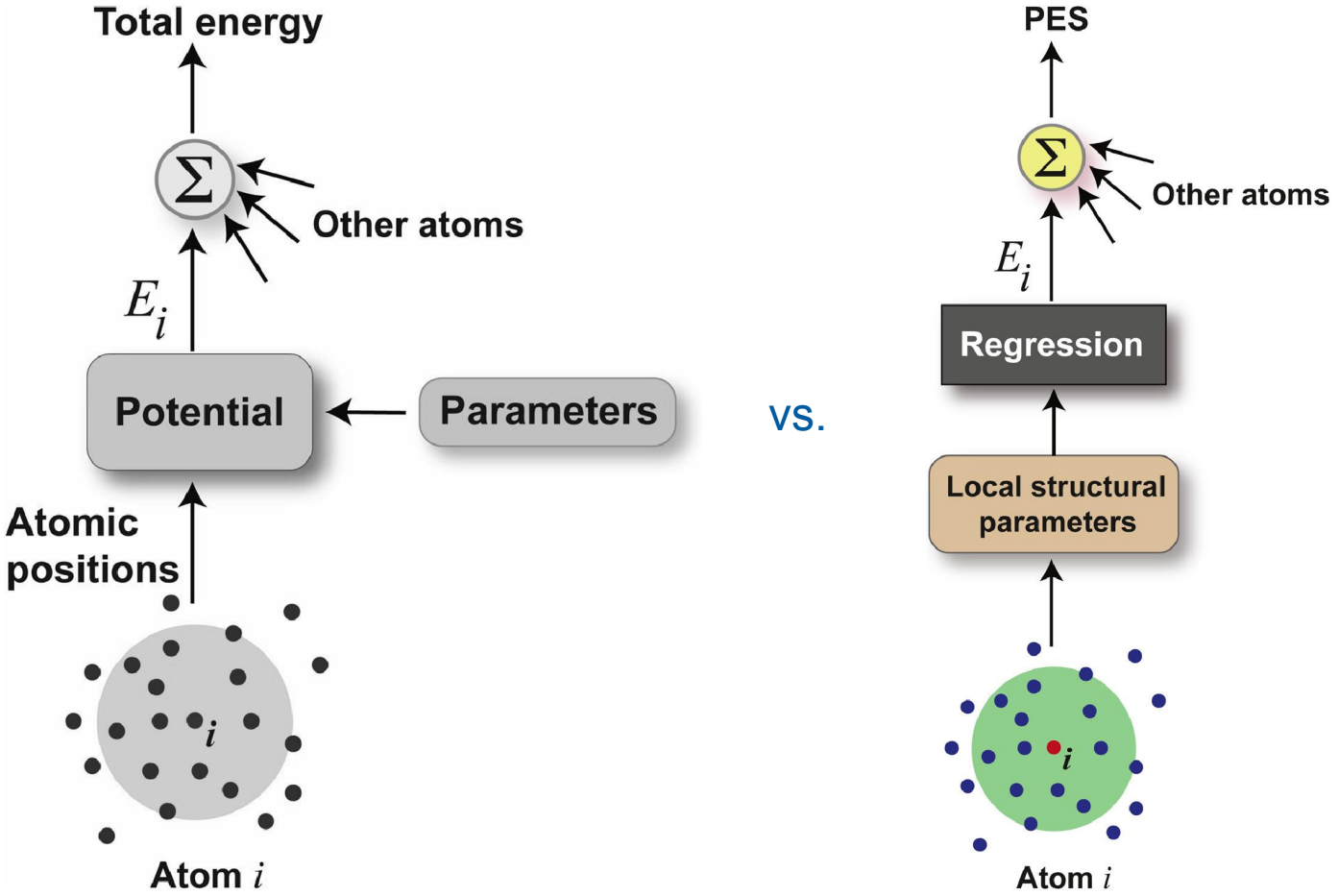
Diese Maßnahme wird mitfinanziert mit Steuermitteln auf Grundlage des vom Sächsischen Landtag beschlossenen Haushaltes.

3.3 Overview of Machine Learning Potentials

Workflow of machine learning potentials

Fitting the PES with machine learning potentials:

- The local environment of an atom within the cutoff sphere (green) is encoded in a set of local structural parameters.
- These which are mapped onto the energy assigned to atom using a regression model.
- The summation of the energies of other atoms of the system gives the total energy and thus a point on the PES of the system.



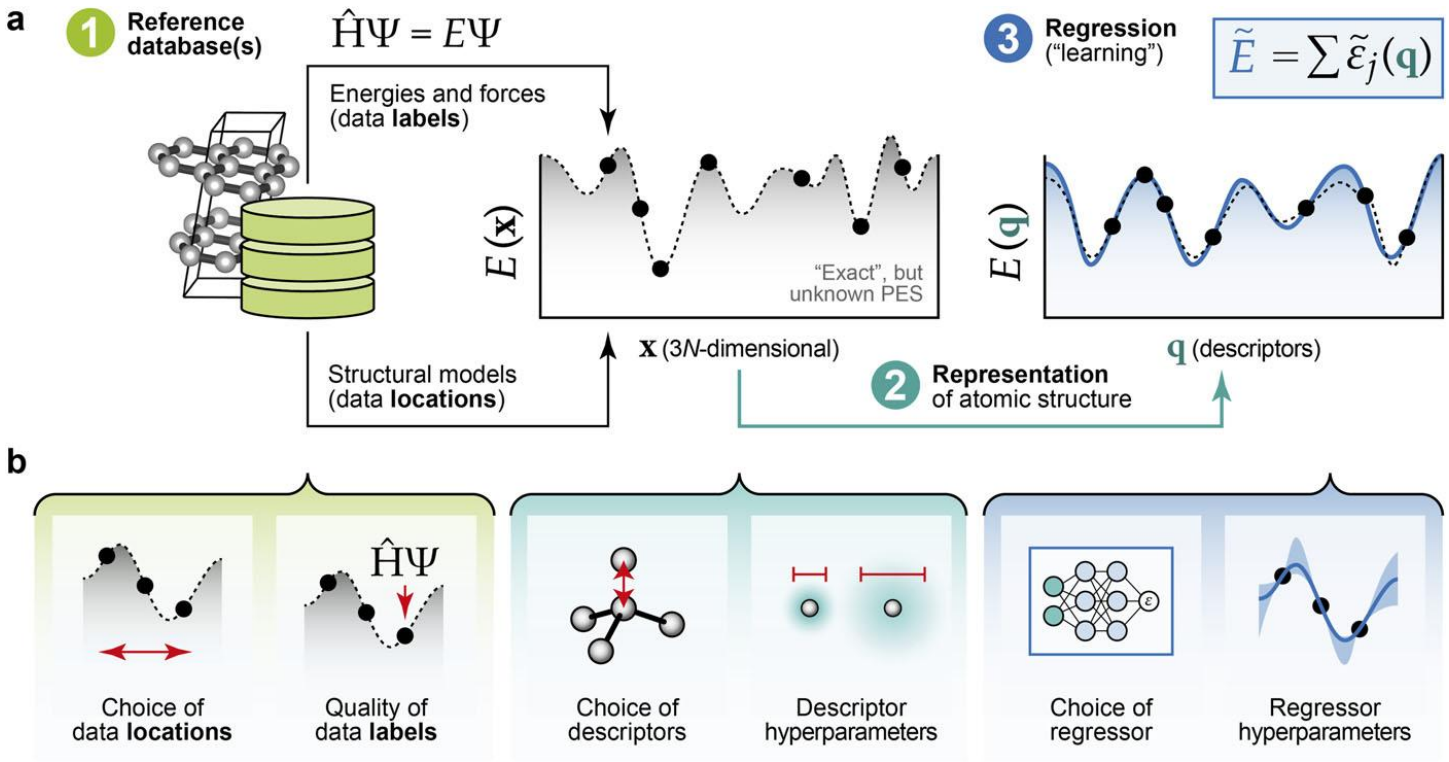
Y. Mishin, Acta Materialia 214, 116980 (2021).

3.3 Overview of Machine Learning Potentials

Workflow of machine learning potentials

Fitting the PES with machine learning potentials:

- The local environment of an atom within the cutoff sphere (green) is encoded in a set of local structural parameters.
- These which are mapped onto the energy assigned to atom using a regression model.
- The summation of the energies of other atoms of the system gives the total energy and thus a point on the PES of the system.



J.D. Morrow et al., J. Chem. Phys. 158, 121501 (2023).

3.3 Overview of Machine Learning Potentials

Types of machine learning potentials

Linear regression potentials

- Polynomial fit of PES

😊 Simple mathematical form
😊 Efficient to evaluate

😕 Require descriptors

SNAP, MTP, ACE

Kernel regression potentials

- Non-linear relation between a pair of random variables

😊 Less data needed
😊 Uncertainty estimates

😕 Scales unfavorably (cubic)

GAP, FLARE

Neural network potentials

- Universal approximation based on nonlinear mapping

😊 Accuracy
😊 Transferability

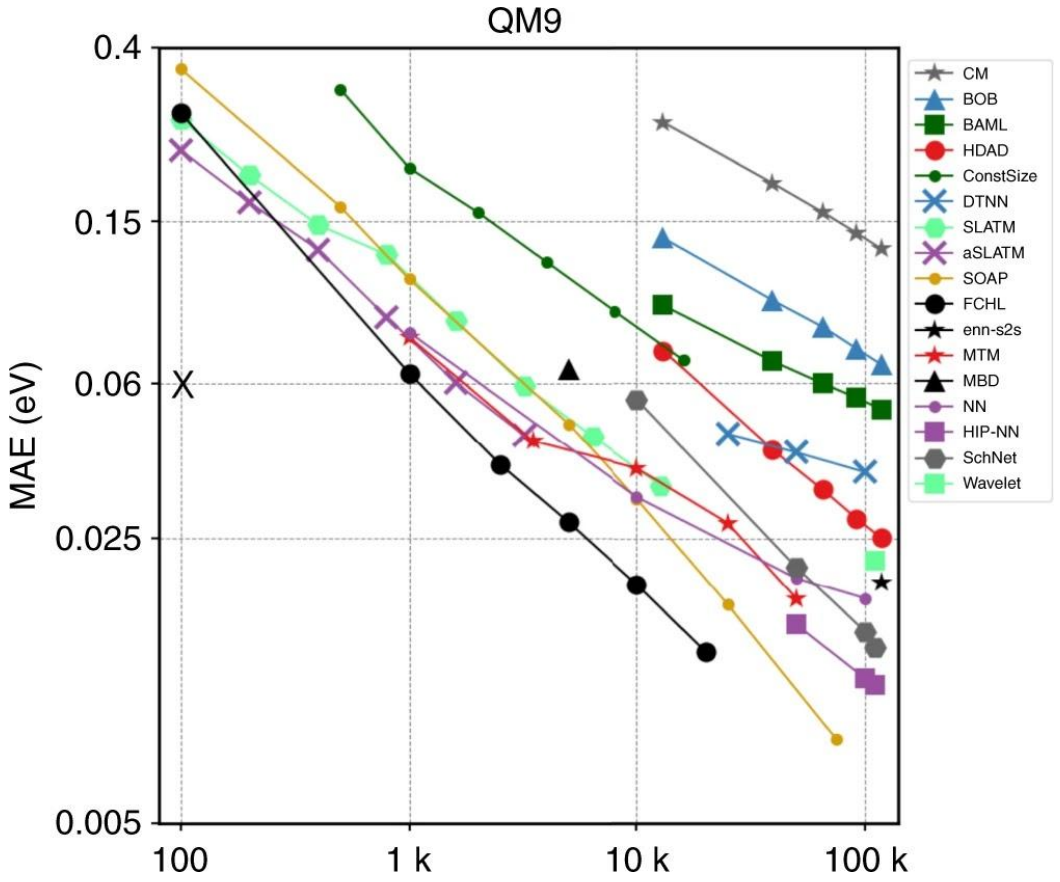
😕 Challenges in training and hyperparameter optimization

Behler-Parrinello, DeepMD, SchNet, NequIP, MACE, Allegro

3.3 Overview of Machine Learning Potentials

Accuracy of machine learning potentials

- Learning curves of atomization energies of organic molecules, showing out-of-sample prediction error (mean absolute error) decays with increasing number of training molecules drawn at random from QM9 dataset.
- Dataset including 134,000 stable small organic molecules made up of C, H, O, N, and F.
- Subset of the 166 billion organic molecules in the GDB-17 database of compounds relevant for drug design and lead compounds.

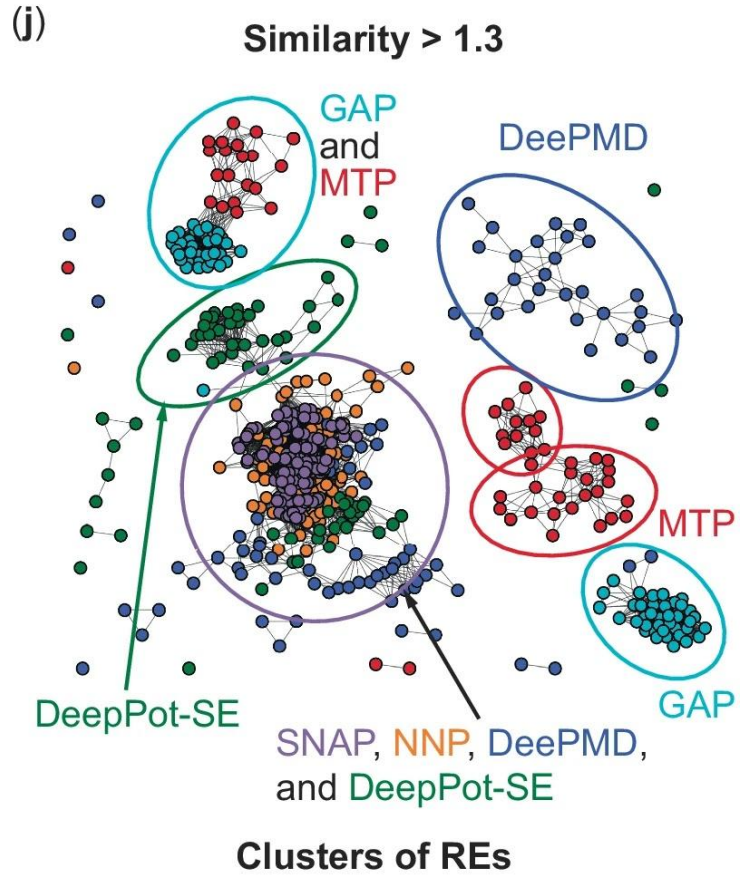


O. A. von Lilienfeld et al., Nature Communications 11, 4895 (2020)

3.3 Overview of Machine Learning Potentials

Accuracy of machine learning potentials

- Benchmarking potentials for wide range of properties including defects, rare event forces, diffusion, phonon, thermal conduction, and elastic modulus.
- Similarities of the error metrics allow the analysis of the joint performances
- Models that are clustered exhibit similar errors on the same properties
- The scattering of DeepMD models in the similarity clustering based on the defect formation energy category suggests that they can only have good predictions in some properties in the defect formation energy but perform poorly in others.

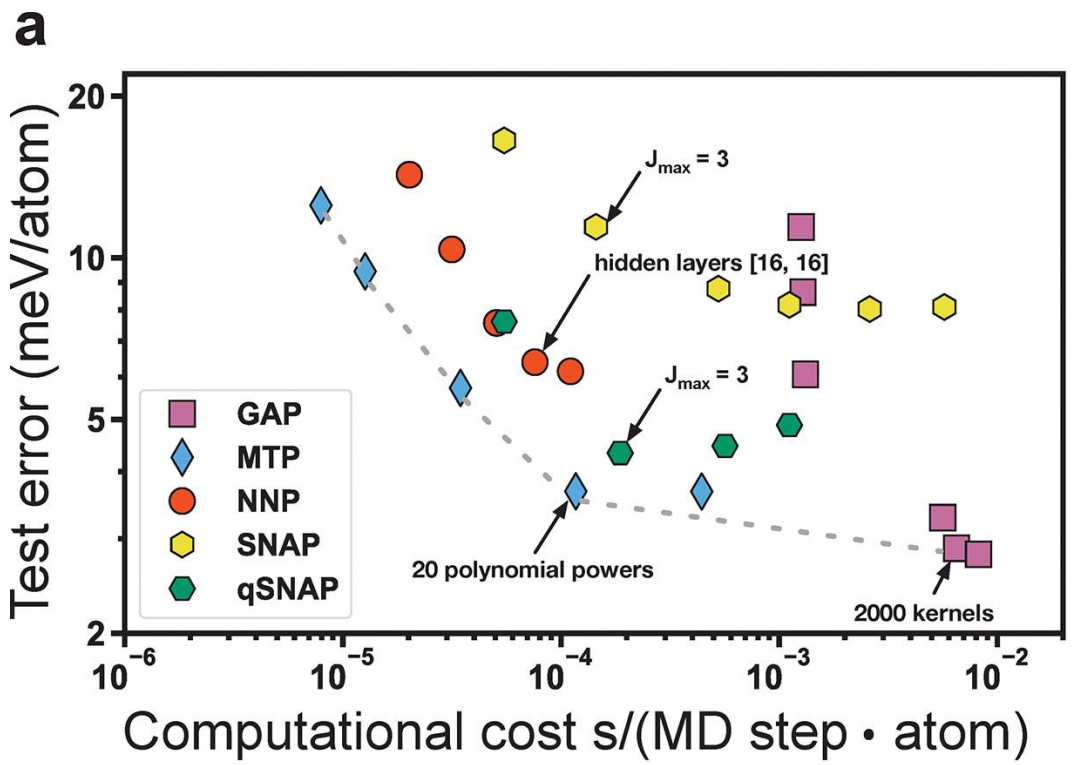


Y. Liu et al, npj Computational Materials 10, 159 (2024).

3.3 Overview of Machine Learning Potentials

Computational performance of machine learning potentials

- Error vs. computational cost for Mo system.
- Dashed line corresponds to the Pareto frontier which is indicates an optimal trade-off between accuracy and computational cost.

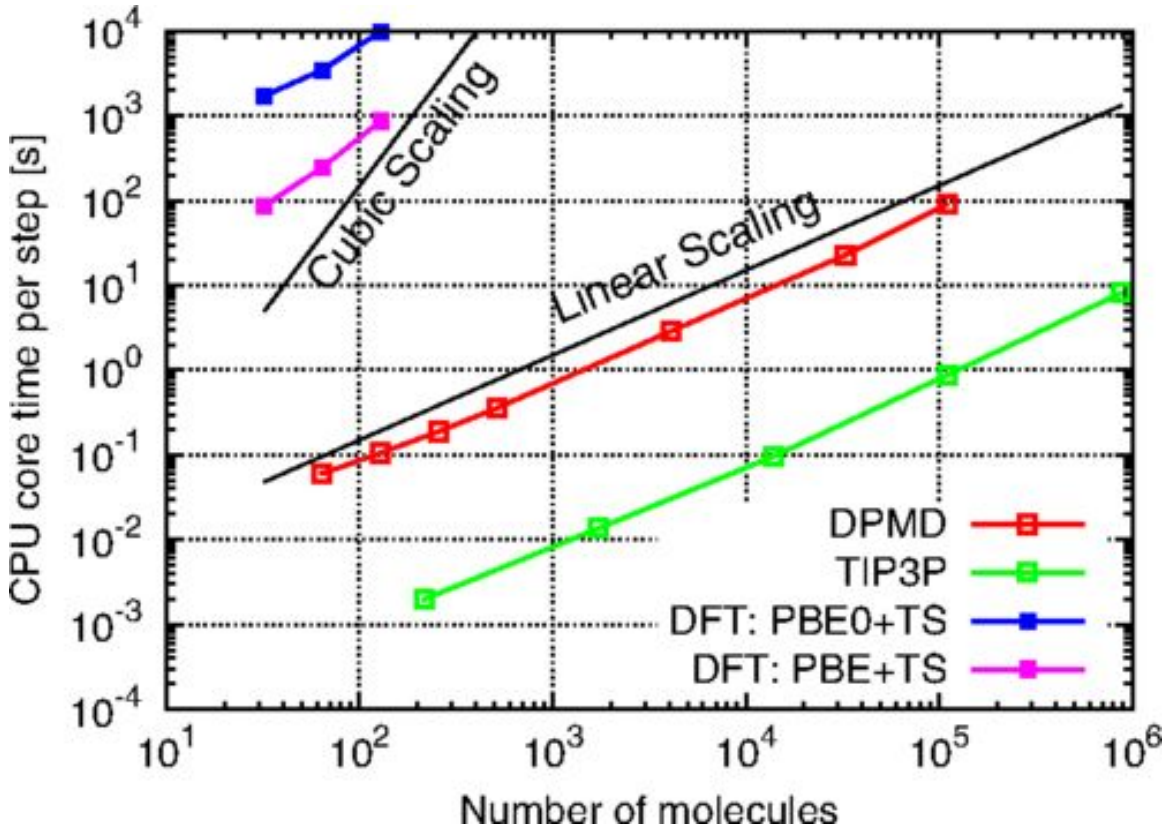


Y. Zuo et al, J. Phys. Chem. A 124, 731 (2020).

3.3 Overview of Machine Learning Potentials

Computational performance of machine learning potentials

- Computational cost for a time-step vs. the number of molecules for liquid water simulations.
- First-principles methods have a cubic scaling with the number of atoms.
- Interatomic potentials (both classical and machine learning) scale linearly.



L. Zhang et al., Phys. Rev. Lett. 120, 143001 2018).

3.4 Data and Descriptors

Methods for generating data

First-principles molecular dynamics (DFT)

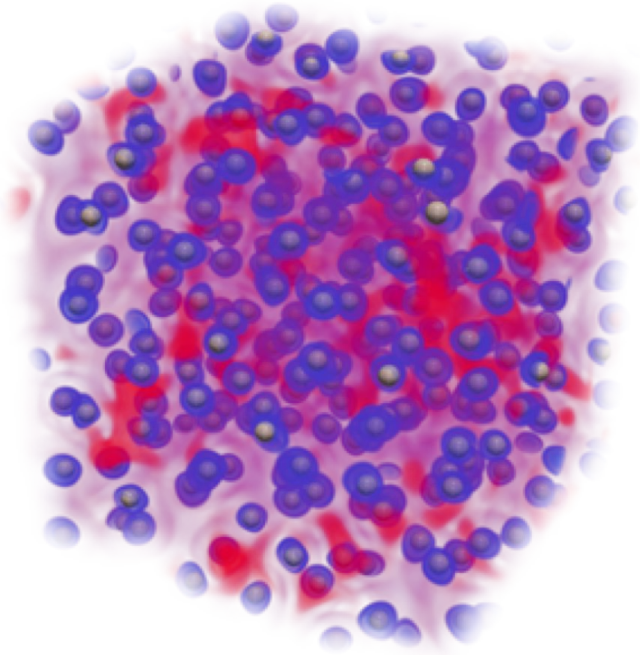
- Structures are generated by simulating atomic motion at a specific temperature and pressure.
- Configurations are saved at regular intervals.

Force field trajectories

- Pre-existing conventional force fields are used to generate initial configurations.
- Potential bias from force field assumptions needs to be considered.

Random atomic configurations

- Atoms or molecules are randomly placed within constraints (e.g., minimum atomic distances).
- This ensures chemically reasonable initial configurations.



3.4 Data and Descriptors

Methods for generating data

Random displacements

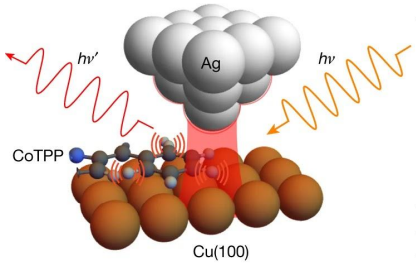
- Atomic displacements are applied to known crystal structures to explore local energy minima.

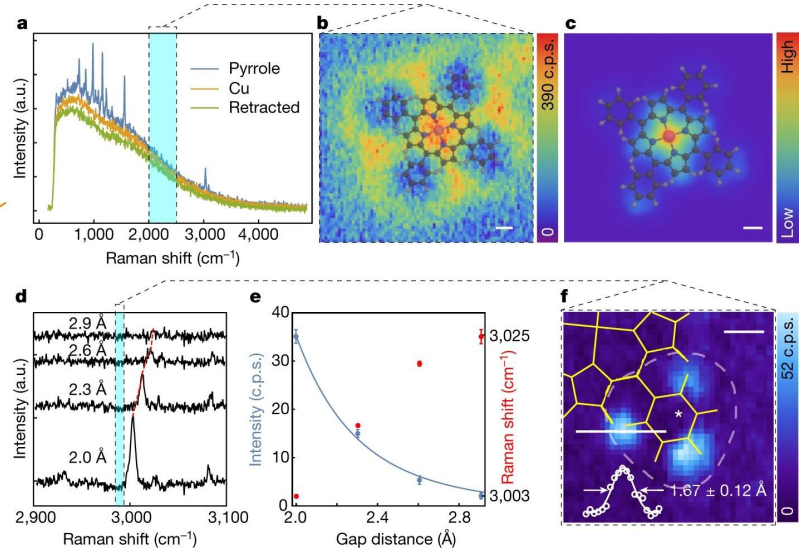
Normal mode sampling

- Commonly used for molecular systems.
- Structures are generated by sampling vibrational normal modes.

Metadynamics

- Enhances the sampling of rare events and transition states by biasing the system's energy landscape.





J. Lee et al., Nature 568, 78 (2019).

3.4 Data and Descriptors

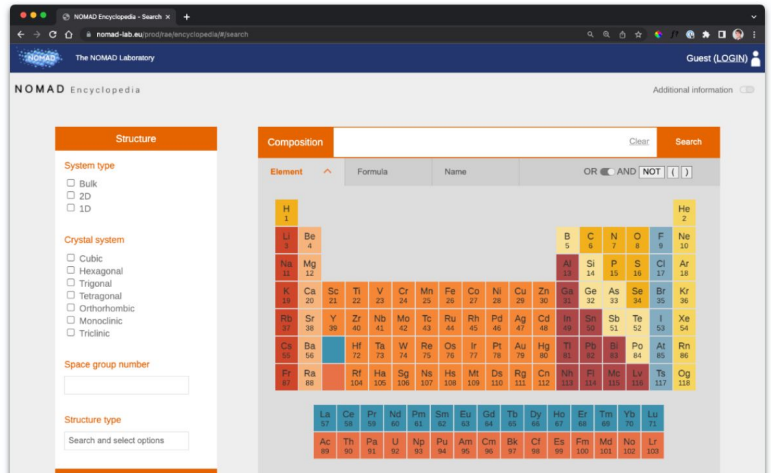
Methods for generating data

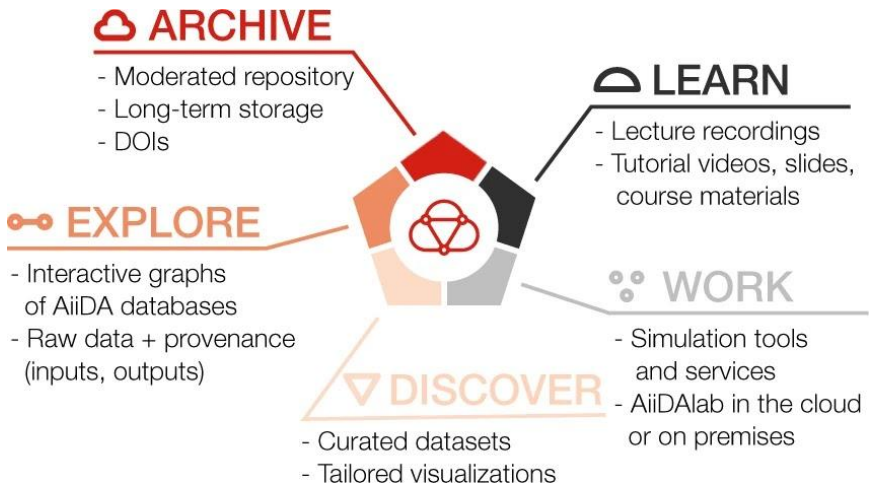
Fragment-based sampling

- Larger systems are decomposed into smaller fragments for more focused sampling.

Database and repository usage

- Public or proprietary databases provide access to pre-computed atomic configurations.
- Care must be taken to ensure compatibility with training requirements (e.g., consistent settings and input formats).





3.4 Data and Descriptors

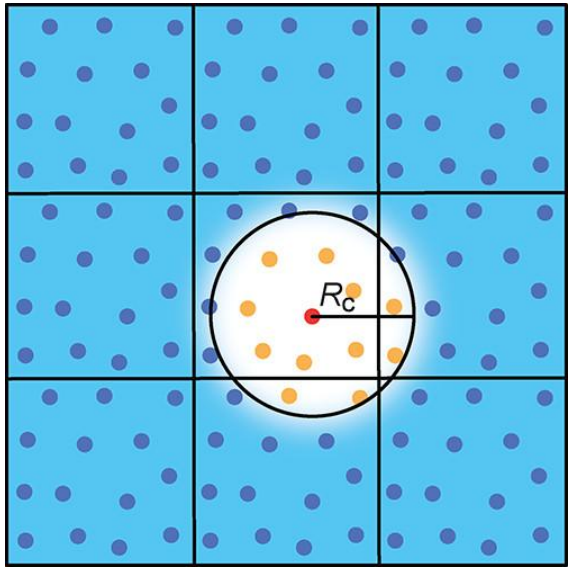
Generating data

Methods for generating data

- The quality of the dataset determines the accuracy and transferability.
- The dataset's reliability depends on the chosen electronic structure method, often DFT.
- Higher-level methods like coupled cluster can provide more accurate results but are computationally expensive.

$$E = \sum_i E_i$$

$$\boldsymbol{R}_i \longrightarrow \boldsymbol{G}_i \xrightarrow{M} E_i$$



J. Behler, Chem. Rev. 121, 10037 (2021).

Composition of the training database

- Typically contains 1,000 to 10,000 snapshots.
- Includes energies, forces, and often stresses, derived from first-principles molecular dynamics (usually DFT).
- Can involve snapshots from first-principles molecular dynamics.

3.4 Data and Descriptors

Generating data

Diversity in Data

- The database should be diverse and representative of configurations relevant to the intended application.

Structure Selection

- Low-energy configurations are prioritized, but non-equilibrium structures (e.g., defects, transition states) are included to ensure coverage of the configuration space.
- For metals, defects like dislocations, grain boundaries, and stacking faults are often emphasized.
- For covalent materials, diverse phases, including liquids and amorphous forms, as well as specific defects, are included.

3.4 Data and Descriptors

Generating data

Role of Human Expertise

- Human input is often crucial for selecting and curating reference structures.
- Expert knowledge ensures relevance to targeted applications but may introduce redundancies or biases.

Challenges and Strategies

- Striking a balance between database completeness and computational manageability.
- Over-reliance on hand-picking can lead to inefficiencies, prompting the adoption of more automated approaches like active learning.

Active learning strategies are an active area of research

3.4 Data and Descriptors

Descriptors

Local Structural Descriptors

- Encodes the local environment of an atom into a fixed-length feature vector.
- Ensures rotational, translational, and permutational invariance, critical for accurate ML models.
- Provides smooth representations of atomic environments to enable effective regression.

$$E = \sum_i E_i$$

$$\boldsymbol{R}_i \longrightarrow \boldsymbol{G}_i \xrightarrow{M} E_i$$

Key Requirements

- **Efficiency:** Must distinguish between different environments while being computationally manageable.
- **Completeness:** Should theoretically reconstruct the atomic environment accurately, avoiding loss of structural information.
- **Fixed Size:** Converts variable-sized position vectors into a fixed-length feature vector, allowing standardized processing in ML frameworks.

3.4 Data and Descriptors

Comprehensive list of descriptors

Atom-Centered Symmetry Functions (ACSFs)

Smooth Overlap of Atomic Positions (SOAP)
- Provides rotational, translational, and permutational invariances.

Coulomb Matrix
- Captures atomic arrangements based on charges and distances.

Bispectrum
- Encodes rotational invariance through spherical harmonics.

Bag of Bonds
- Represents molecules as sets of interatomic bond properties.

Ewald Sum Matrix
- Includes long-range electrostatics for molecular representations.

Overlap Matrix-Based Descriptors
- Derived from quantum chemistry overlap integrals.

Faber-Christensen-Huang-Lilienfeld (FCHL) Descriptors
- Built for systematic scalability and transferability.

Many-Body Tensor Representation (MBTR)
- Encodes high-order structural correlations.

Spherical Harmonics-Based Descriptors
- Extend bispectrum-like methods.

Weighted Symmetry Functions
- Enhance ACSFs by emphasizing specific local features.

Polynomials in Moment Tensor Potentials (MTPs)
- Efficiently encode structural configurations.

Histogram of Internal Coordinates
- Simplifies representation by binning distances and angles.

Chebyshev Polynomials
- Provide compact representations for certain systems.

Long-Distance Equivariant Representations
- Account for extended interactions while maintaining symmetry properties.

Smoothed Atomic Densities
- Capture continuous atomic density distributions.

Permutation Invariant Polynomials
- Ensure symmetry in small molecular systems.

Attila Cangi | a.cangi@hzdr.de

3.4 Data and Descriptors

Descriptors

Common types of descriptors

- **Gaussian symmetry functions (Behler-Parrinello type)**
  Represent pairwise and angular dependencies.
- **Moment tensor descriptors**
  Provide higher-order interactions using tensor contractions.
- **SOAP (Smooth Overlap of Atomic Positions)**
  Uses density expansions for environment representation but is computationally intensive.
- **Zernike polynomials**
  Compact and orthogonal representation of environments.
- **Atomic Cluster Expansion (ACE)**
  Generates invariant polynomials for multicomponent systems.

Trade-offs in descriptor design

- Balancing completeness and computational cost.
- Overcomplete descriptors may lead to redundancy and discontinuities.

3.5 Examples of Machine Learning Potentials

Linear regression potential

- Atomic cluster expansion

Regression kernel potential

- Gaussian approximation potential

Neural network potential

- Behler-Parrinello neural network potential

Deep neural network and advanced potentials

- MACE potential

3.5.1 Example 1: Atomic cluster expansion

Many-atom expansion

- Known expression for expanding the total energy into contributions from sites energies.
- Even if one term can be evaluated fast (linearly), the combinatorial scaling of the multi-index sums makes this impractical for large systems.

$$E_i = V_0(Z_i) + \sum_{j_1} V_1(r_{ij_1}, Z_{j_1}; Z_i) + \sum_{j_1<j_2} V_2(r_{ij_1}, Z_{j_1}, r_{ij_2}, Z_{j_2}; Z_i) + \ldots + \sum_{j_1<\cdots<j_{\bar{N}}} V_{\bar{N}}(r_{ij_1}, Z_{j_1}, \ldots, r_{ij_{\bar{N}}}, Z_{j_{\bar{N}}}; Z_i)$$

Atomic cluster expansion

- Reorder many-atom expansion and consider repeated and spurious (self-interaction) clusters.

$$E_i = U_0(Z_i) + \sum_{j_1} U_1(r_{ij_1}, Z_{j_1}; Z_i) + \sum_{j_1,j_2} U_2(r_{ij_1}, Z_{j_1}, r_{ij_2}, Z_{j_2}; Z_i) + \ldots + \sum_{j_1,\ldots,j_{\bar{N}}} U_{\bar{N}}(r_{ij_1}, Z_{j_1}, \ldots, r_{ij_{\bar{N}}}, Z_{j_{\bar{N}}}; Z_i)$$

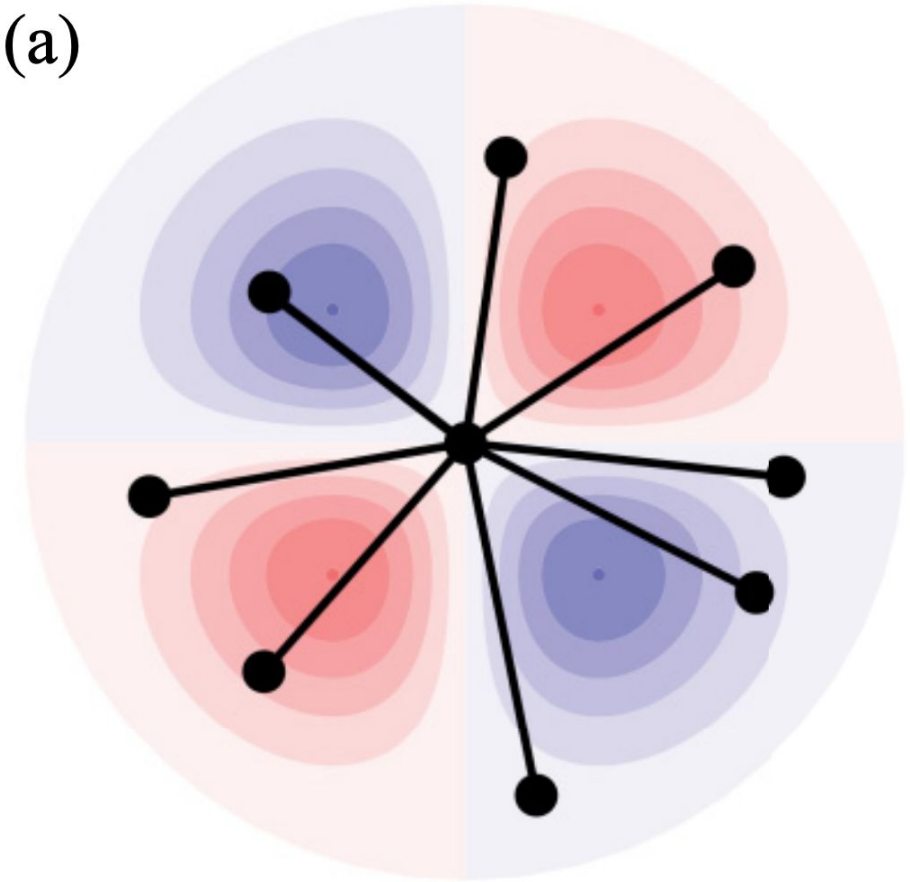
3.5.1 Example 1: Atomic cluster expansion

Many-atom expansion

- Known expression for expanding the total energy into contributions from sites energies.
- Even if one term can be evaluated fast (linearly), the combinatorial scaling of the multi-index sums makes this impractical for large systems.

Atomic cluster expansion

- Reorder many-atom expansion and consider repeated and spurious (self-interaction) clusters.

E (a)  $+\ldots$

E $..$

R. Drautz, Phys. Rev. B 99, 014104 (2019).

3.5.1 Example 1: Atomic cluster expansion

Simple four-stage evaluation scheme

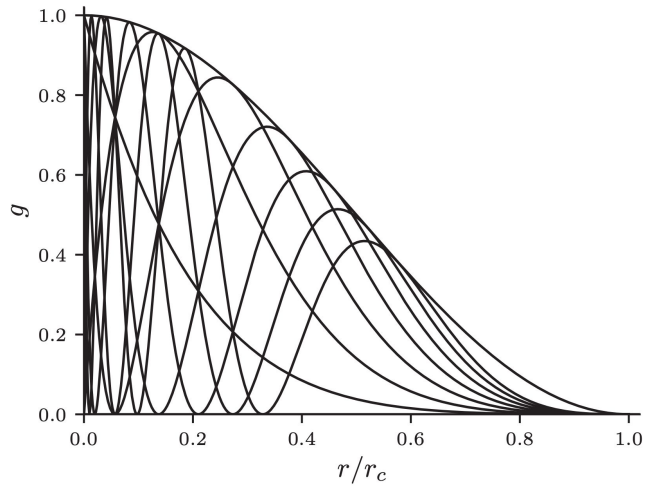
- The evaluation cost scales only linearly in the number of neighbours, unlike the combinatorial scaling of a naive cluster expansion.

$$E_i = \boldsymbol{\theta} \cdot \boldsymbol{B}^{(i)}$$

$$\boldsymbol{B}^{(i)} = C\, \boldsymbol{A}^{(i)}$$

$$\boldsymbol{A}^{(i)}_{\boldsymbol{nlm}} = \prod_{t=1}^{\bar{N}} A^{(i)}_{n_t l_t m_t}$$

$$A^{(i)}_{nlm} = \sum_j R_{nl}(r_{ij}, Z_j, Z_i)\, Y_l^m(\hat{\boldsymbol{r}}_{ij})$$



R. Drautz, Phys. Rev. B 99, 014104 (2019).

3.5.1 Example 1: Atomic cluster expansion

- **Forces** are obtained by differentiation
  - Involves gradients of functions of the atomic density expanded in an atomic cluster expansion

$$\boldsymbol{F}_k = -\nabla_k E = -\nabla_k \sum_i F(\rho_i^{(p)})$$

$$\boldsymbol{F}_k = -\sum_{inlm} \sum_p \sum_{K\boldsymbol{nl}} c_{\boldsymbol{nl}}^{(K,p)} \frac{\partial F}{\partial \rho_i^{(p)}} \frac{\partial B_{\boldsymbol{inl}}^{(K)}}{\partial A_{inlm}} \nabla_k A_{inlm}$$

- **Training and loss function**
  - Obtain an error that approximately scales with the energy of the reference data.

$$L = \sum_n w_n (E_n - E_n^{ref})^2 + \alpha_1 \sum_{K\boldsymbol{nl}} |c_{\boldsymbol{nl}}^{(K)}| + \alpha_2 \sum_{K\boldsymbol{nl}} |c_{\boldsymbol{nl}}^{(K)}|^2$$

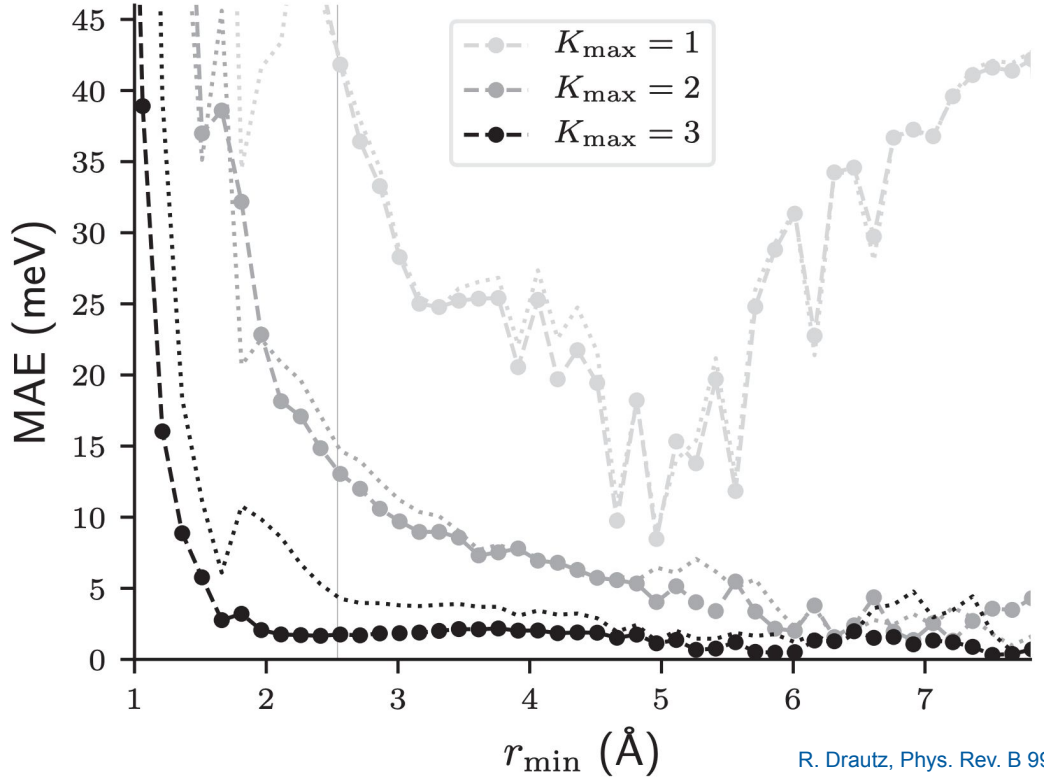
$$w_n = \omega / \left[E_n^{ref} - E_{min}^{ref} (1 + \Delta) \right]^2$$

Attila Cangi | a.cangi@hzdr.de

3.5.1 Example 1: Atomic cluster expansion

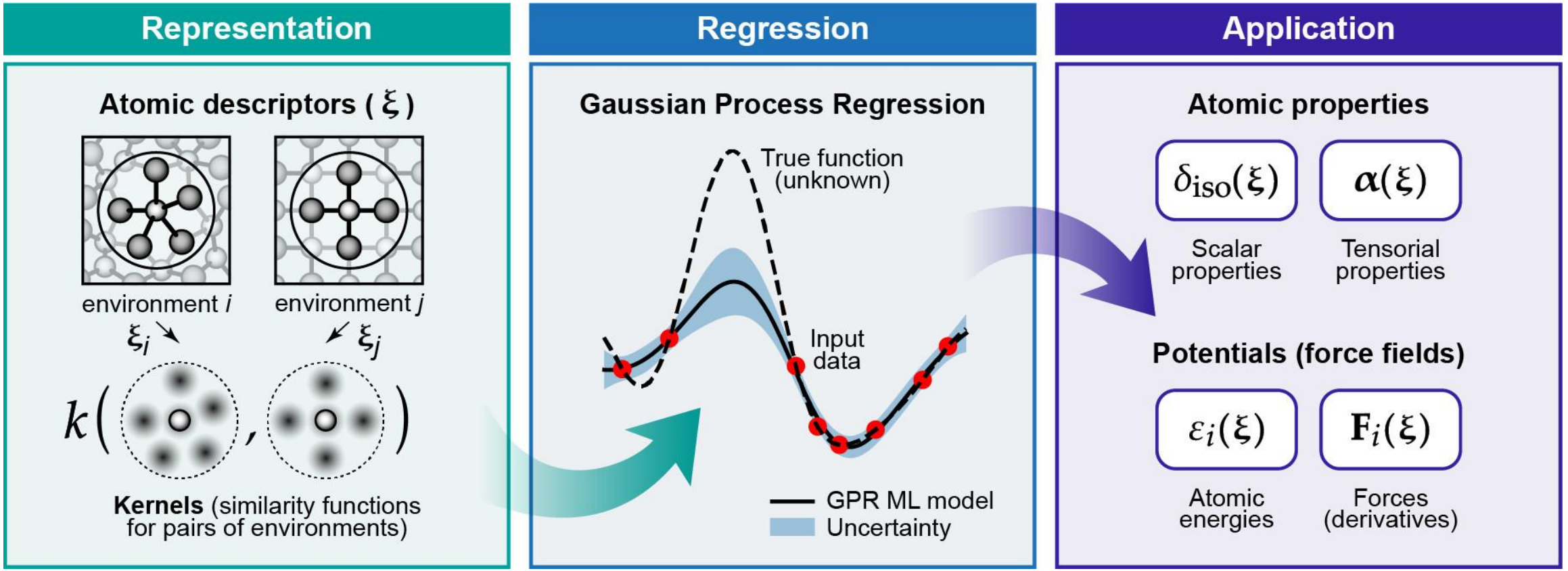
Machine learning potential for copper

- The error in total energy converges quickly with the order of the expansion.



R. Drautz, Phys. Rev. B 99, 014104 (2019).

3.5.2 Example 2: Gaussian approximation potential (GAP)



V. L. Deringer et al., Chem. Rev. 121, 10073 (2021).

3.5.2 Example 2: Gaussian approximation potential (GAP)

Introduction to Gaussian process regression (GPR)

- GPR is nonlinear, nonparametric regression for interpolating between data points scattered in a high-dimensional input space.
- Nonparametric regression does not assume a closed functional form.
- Instead, we fit a flexible function based on a large amount of data.
- Once fit function has been “trained” we can evaluate it to make predictions.
- Consider a smooth, regular function, y, which takes a d-dimensional vector as input and maps it onto a single scalar value:

$$y \ : \ \mathbb{R}^d \rightarrow \mathbb{R}$$

- We do not know the functional form of y, but we have made N independent observations, y_n , of its value at the locations x_n , resulting in a dataset:

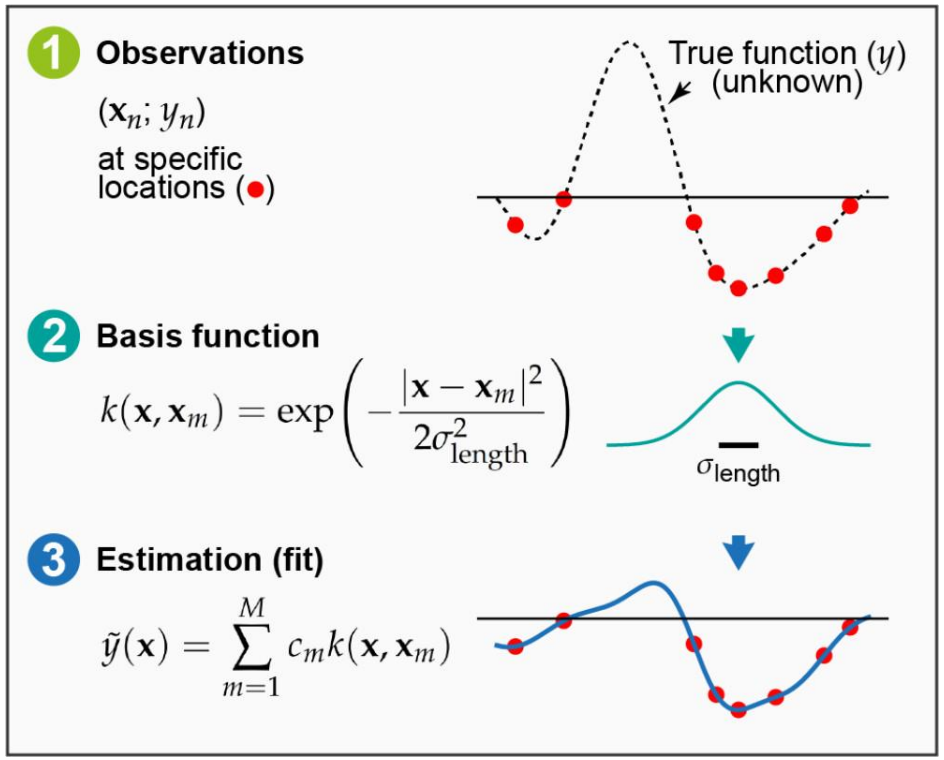
$$\mathcal{D} = \{\boldsymbol{x}_n; y_n\}_{n=1}^{N}$$

- The goal is now to use these data values to create an estimator that can predict the continuous function y at arbitrary locations x and also to quantify the uncertainty (“expected error”) of this prediction.

3.5.2 Example 2: Gaussian approximation potential (GAP)

Introduction to Gaussian process regression (GPR)

- We find an approximation defined as a linear combination of M basis functions with corresponding coefficients.
- The basis functions are placed at arbitrary locations in the input space.
- Properties of the function k:
  - describes the similarity when evaluated at two arbitrary locations.
  - is symmetric to swapping its arguments.
  - is positive semidefinite.

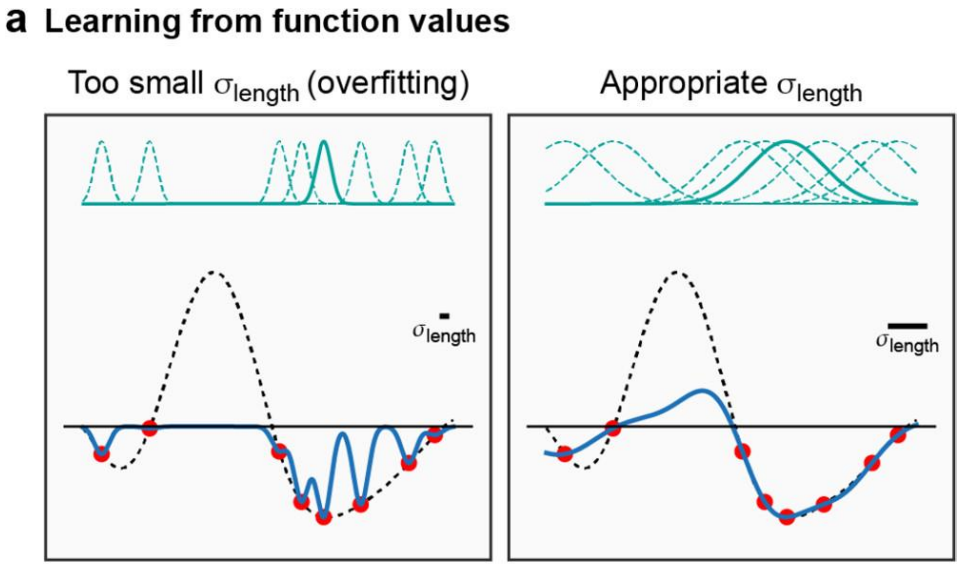


V. L. Deringer et al., Chem. Rev. 121, 10073 (2021).

3.5.2 Example 2: Gaussian approximation potential (GAP)

Introduction to Gaussian process regression (GPR)

- The kernel function does not matter in principle.
- However practical success or failure of a GPR model will depend to a large extent on choosing the appropriate kernel.
- Consider a Gaussian kernel which includes a length scale hyperparameter.
- This kernel is a universal approximator for any setting of the length scale, but choosing an inappropriate length scale will result in very slow convergence as a function of the number of training data points.



V. L. Deringer et al., Chem. Rev. 121, 10073 (2021).

3.5.2 Example 2: Gaussian approximation potential (GAP)

Introduction to Gaussian process regression (GPR)

- Fitting a GPR model means finding the coefficients that minimize the loss function

$$\mathcal{L} = \sum_{n=1}^{N} \frac{\left[y_n - \tilde{y}(\boldsymbol{x}_n)\right]^2}{\sigma_n^2} + R$$

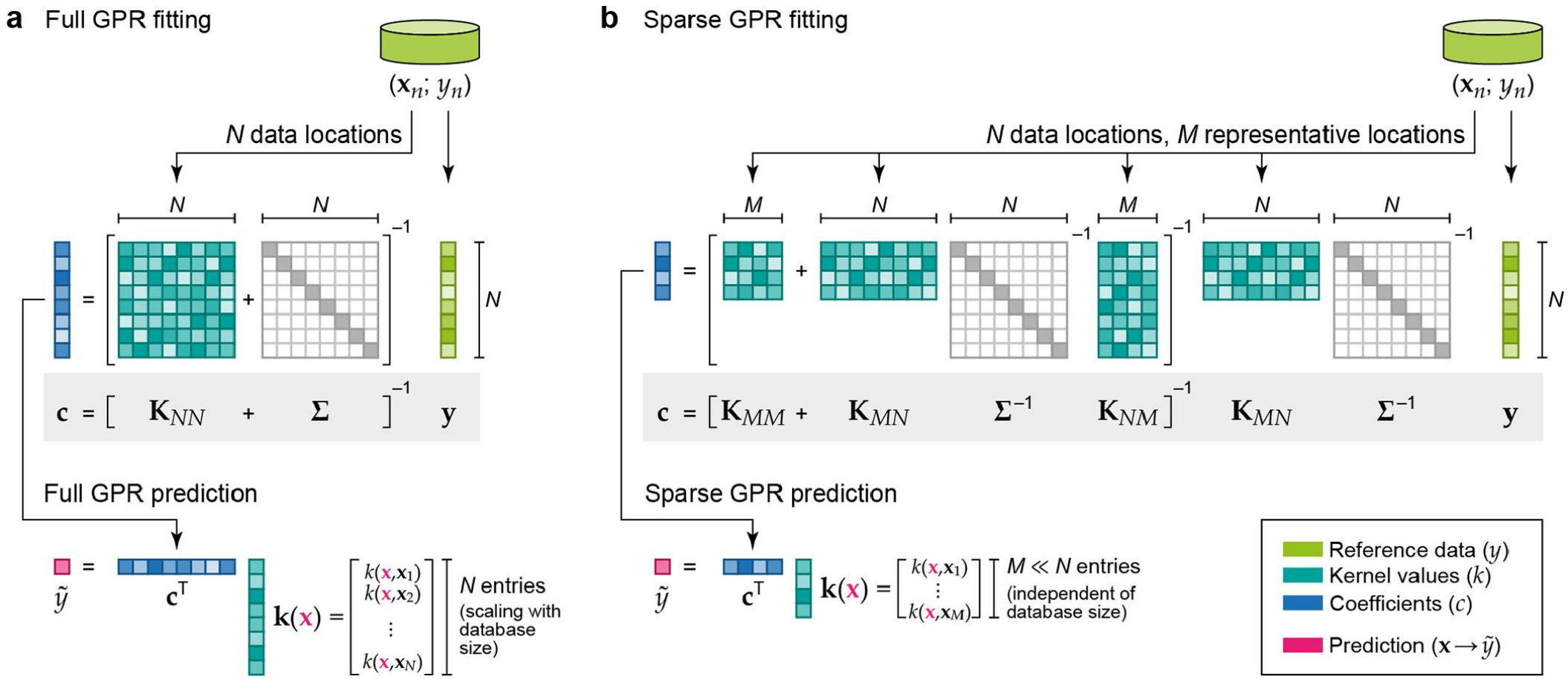
- With a regularization term

$$R = \sum_{m,m'}^{M} c_m \, k(\boldsymbol{x}_m, \boldsymbol{x}_{m'}) \, c_{m'}$$

- Two objectives are included in the loss function.
  - The first term achieves a close fit to the data points, but would alone lead to overfitting.
  - The regularization forces the coefficients to remain small.
  - The collection of hyperparameters adjusts the balance between accurately reproducing the fitting data points and the overall smoothness of the estimator.

3.5.2 Example 2: Gaussian approximation potential (GAP)

Introduction to Gaussian process regression (GPR)



V. L. Deringer et al., Chem. Rev. 121, 10073 (2021).

3.5.2 Example 2: Gaussian approximation potential (GAP)

- GAP learns the PES as a function of the atomic positions from precomputed reference data (usually from DFT).
- GAP performs a non-parametric kernel regression of the PES for estimating a local energy of an atomic environment.
- GAP rewrites the PES (a nonlinear function of the atomic positions) as a linear function in the kernels.
- This enables the use linear algebra to obtain the fitting coefficients during the training stage.

$$E = \sum_i E_i = \sum_i E(\boldsymbol{\xi}_i)$$

$$E(\boldsymbol{\xi}_i) = \sum_n \alpha_n k(\boldsymbol{\xi}_i, \boldsymbol{\xi}_n)$$

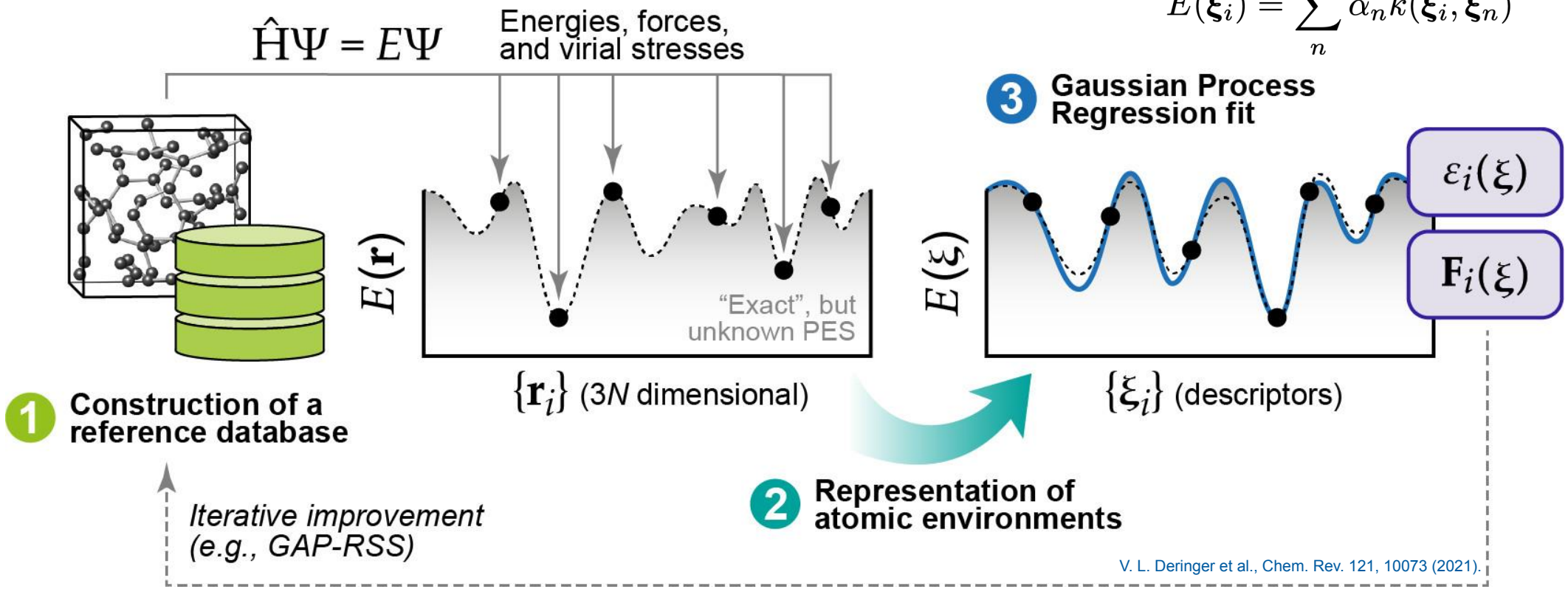
Attila Cangi | a.cangi@hzdr.de

3.5.2 Example 2: Gaussian approximation potential (GAP)

Overview of constructing a GAP potential

$$E = \sum_i E_i = \sum_i E(\boldsymbol{\xi}_i)$$

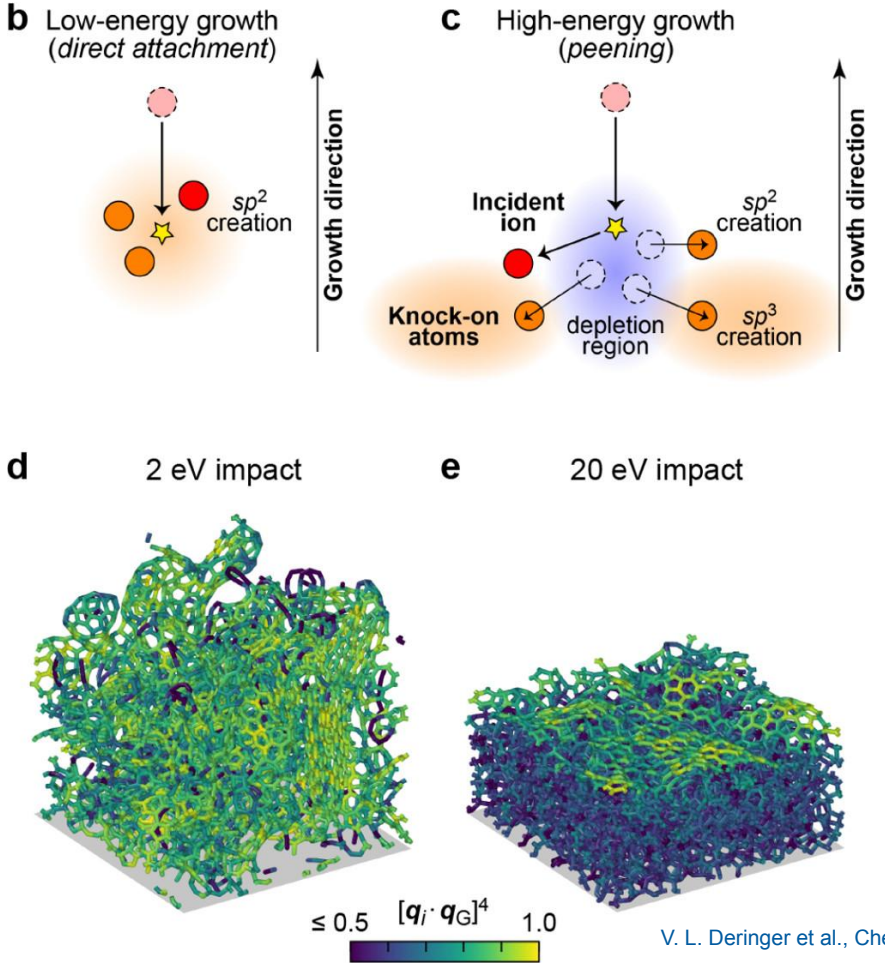
$$E(\boldsymbol{\xi}_i) = \sum_n \alpha_n k(\boldsymbol{\xi}_i, \boldsymbol{\xi}_n)$$



V. L. Deringer et al., Chem. Rev. 121, 10073 (2021).

3.5.2 Example 2: Gaussian approximation potential (GAP)

Application: Amorphous carbon films



V. L. Deringer et al., Chem. Rev. 121, 10073 (2021).

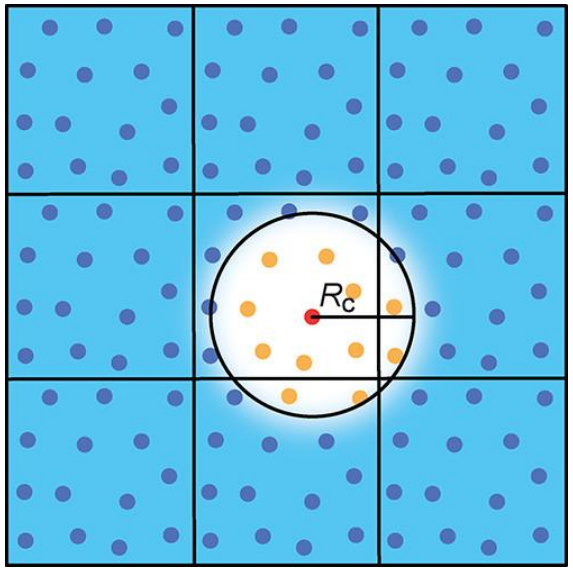
3.5.2 Example 3: Behler-Parrinello Neural Network Potentials (BP)

- Assuming locality, write the PES as a sum of individual atomic energy contributions

$$E = \sum_{i}^{N_{\text{atoms}}} E_i = \sum_{j=1}^{N_{\text{elements}}} \sum_{i=1}^{N_{\text{atoms}}^{j}} E_i^j$$

$$\boldsymbol{R}_i \longrightarrow \boldsymbol{G}_i \xrightarrow{M} E_i$$

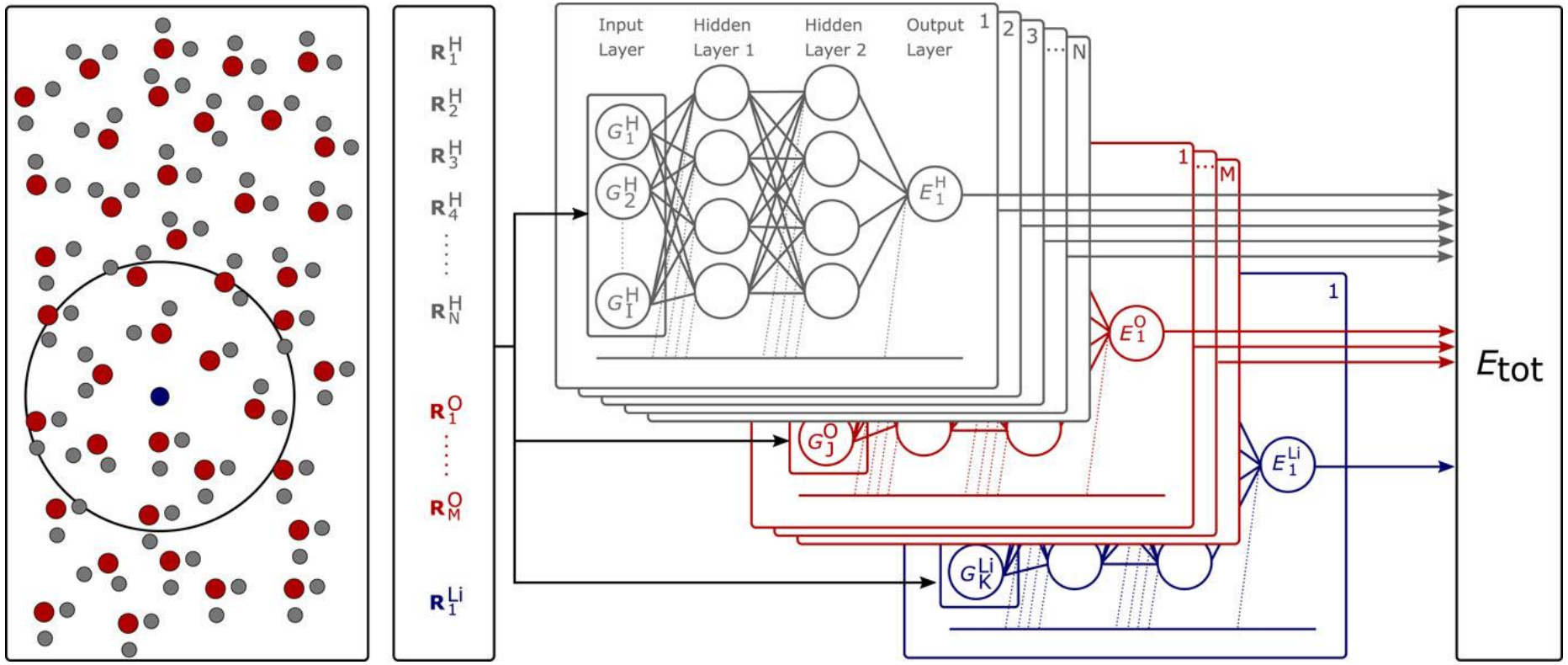
- Coordinates of the atoms are transformed to vectors of atom-centered symmetry functions (ACSF).
- They describe the local atomic environments up to the cutoff radius.
- For each atom, the respective ACSF vector is then used as input for an atomic neural network (NN) predicting its atomic energy.
- Finally, the atomic energies are summed to obtain the total PES.



J. Behler, Chem. Rev. 121, 10037 (2021).

3.5.2 Example 3: Behler-Parrinello Neural Network Potentials (BP)

Example for a lithium hydroxide ion pair in water



A. M. Tokita et al., J. Chem. Phys. 159, 121501 (2023).

3.5.2 Example 3: Behler-Parrinello Neural Network Potentials (BP)

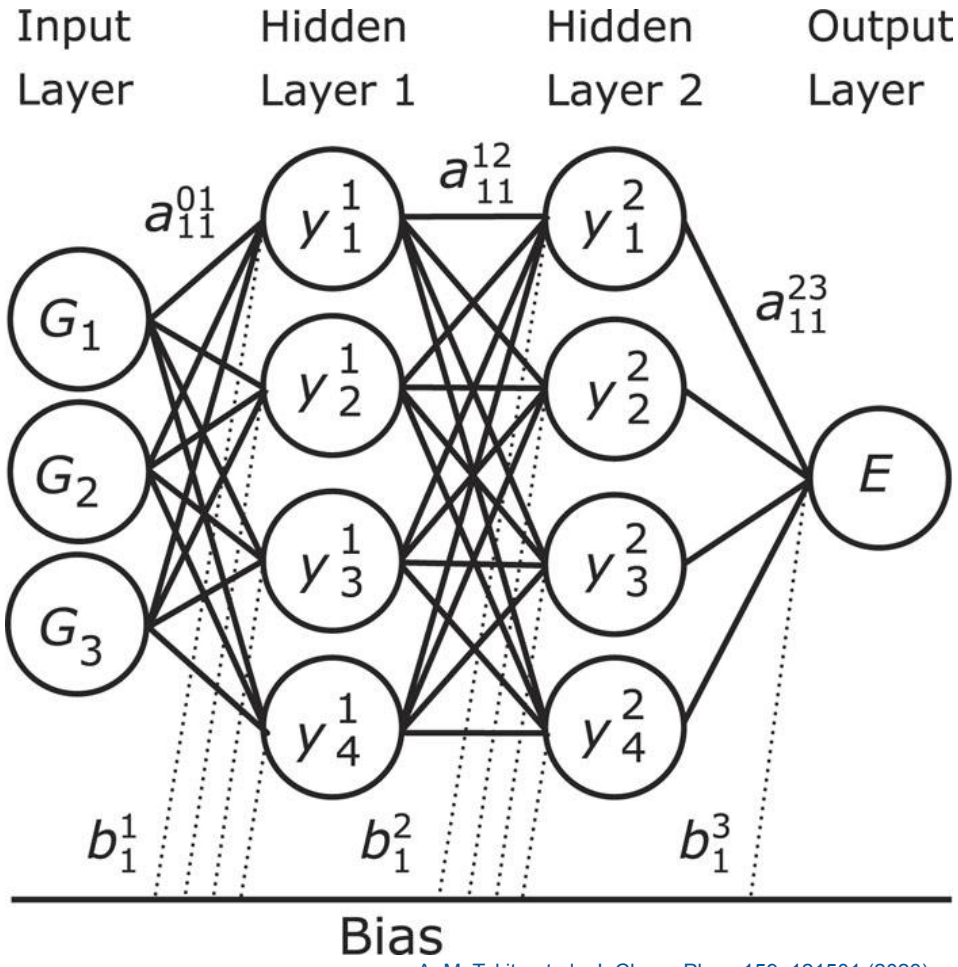
Minimal example

- The numerical value a given node in a given layer is

$$y_j^l = f_j^l \left(b_j^l + \sum_i^{N_{\text{nodes}}^k} a_{ij}^{kl} \cdot y_i^k \right)$$

- The output this small feed-forward neural network is

$$E = f_1^3 \left(b_1^3 + \sum_{k=1}^{4} a_{k1}^{23} \cdot f_k^2 \left(b_k^2 + \sum_{j=1}^{4} a_{jk}^{12} \cdot f_j^1 \left(b_j^1 + \sum_{i=1}^{3} a_{ij}^{01} \cdot G_i \right) \right) \right)$$



A. M. Tokita et al., J. Chem. Phys. 159, 121501 (2023).

3.5.2 Example 3: Behler-Parrinello Neural Network Potentials (BP)

Atom-centered symmetry functions

Each atom is represented by a set of cutoff functions, radial symmetry functions, and angular symmetry functions.

Requirements of suitable atom-centered symmetry functions

- They need to describe the structural details of the atomic environments.
- They must be invariant under rotation, translation, and permutation.
- They must decay smoothly to zero in value and slope at the cutoff radius.
- They need to be continuous and differentiable.
- Their number in the neural network input vectors must be independent of the number of neighbors inside the cutoff sphere.

3.5.2 Example 3: Behler-Parrinello Neural Network Potentials (BP)

Atom-centered symmetry functions

Each atom is represented by a set of:

angular symmetry functions

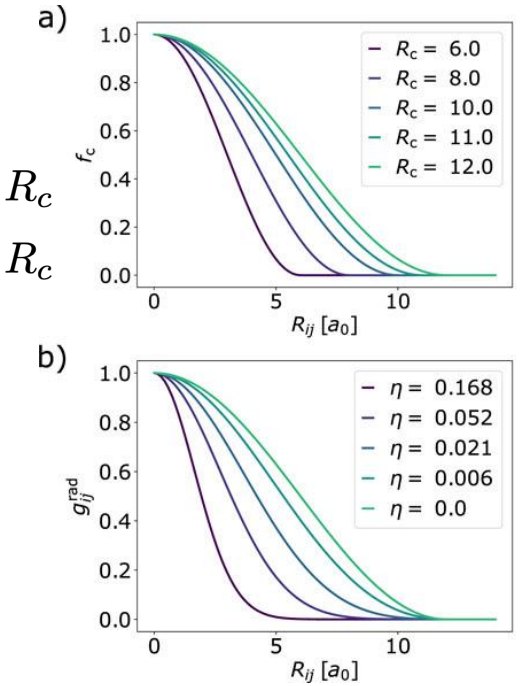
$$G_i^{\text{ang}} = 2^{1-\zeta} \sum_{j \neq i}^{N_{\text{atoms}} \in R_c} \sum_{k \neq i,j}^{N_{\text{atoms}} \in R_c} \left[(1 + \lambda \cos\theta_{ijk})^{\zeta} \; e^{-\eta(R_{ij}^2 + R_{ik}^2 + R_{jk}^2)} \; f_c(R_{ij}) \, f_c(R_{il}) \, f_c(R_{jk}) \right]$$

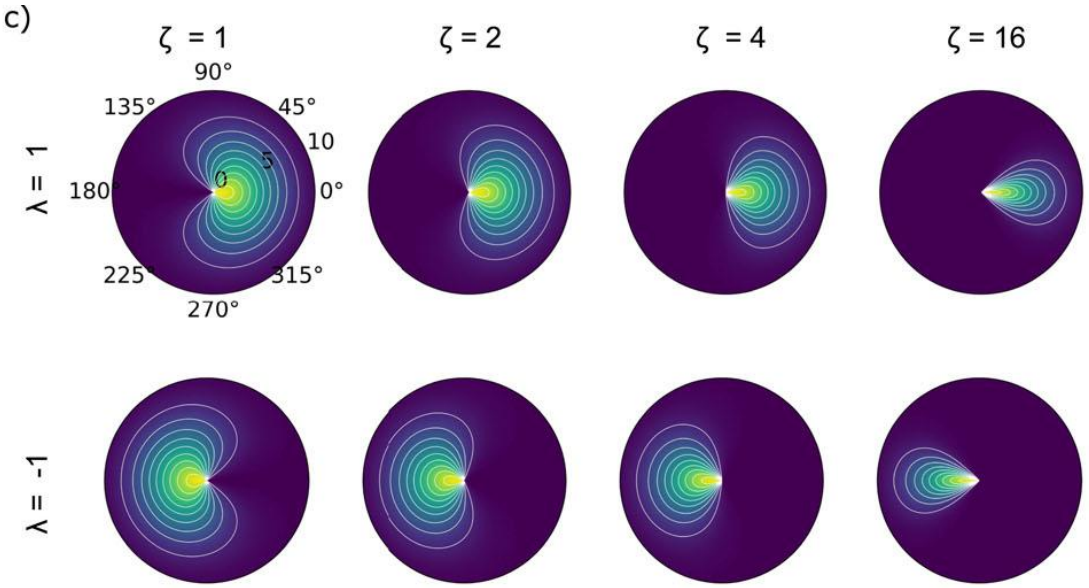
cutoff functions

$$f_c(R_{ij}) = \begin{cases} 0.5 \left[\cos\left(\frac{\pi R_{ij}}{R_c} \right) + 1 \right] & \text{for } R_{ij} \leq R_c \\ 0 & \text{for } R_{ij} > R_c \end{cases}$$

radial symmetry functions

$$G_i^{\text{rad}} = \sum_{j}^{N_{\text{atoms}} \in R_c} e^{-\eta(R_{ij} - R_S)^2} \, f_c(R_{ij})$$





A. M. Tokita et al., J. Chem. Phys. 159, 121501 (2023).

3.5.2 Example 3: Behler-Parrinello Neural Network Potentials (BP)

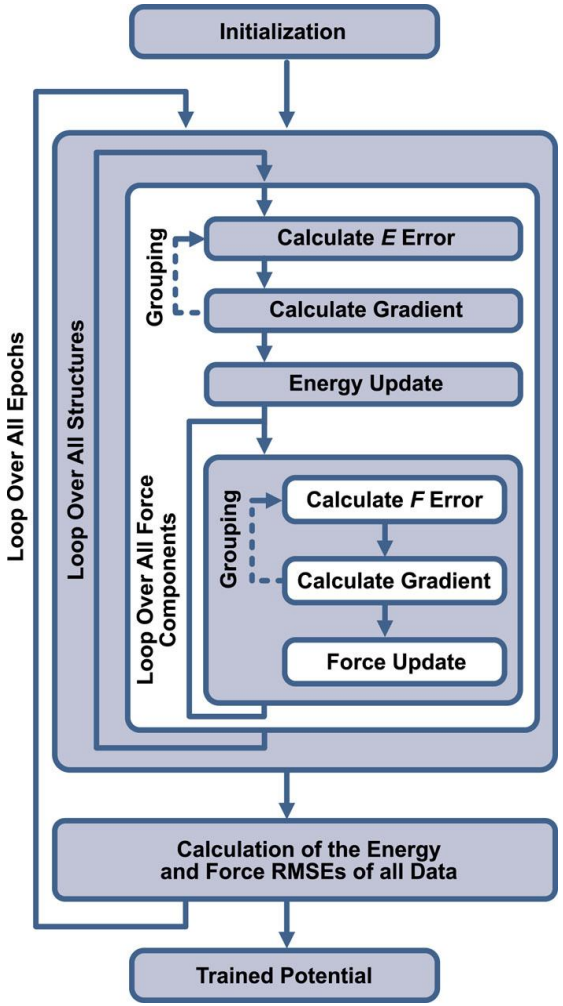
Training and optimization

- Weights are updated once per energy and per force component.
- At the end of each epoch, the RMSEs of the entire training and also test sets are computed.

$$E_{\mathrm{RMSE}} = \sqrt{\frac{1}{N_{\mathrm{struct}}} \sum_{i=1}^{N_{\mathrm{struct}}} \left(E_i - E_{i,\mathrm{BP}}\right)^2}$$

$$\boldsymbol{F}_{\mathrm{RMSE}} = \sqrt{\frac{1}{N_{\mathrm{struct}}} \sum_{i=1}^{N_{\mathrm{struct}}} \left(\boldsymbol{F}_i - \boldsymbol{F}_{i,\mathrm{BP}}\right)^2}$$

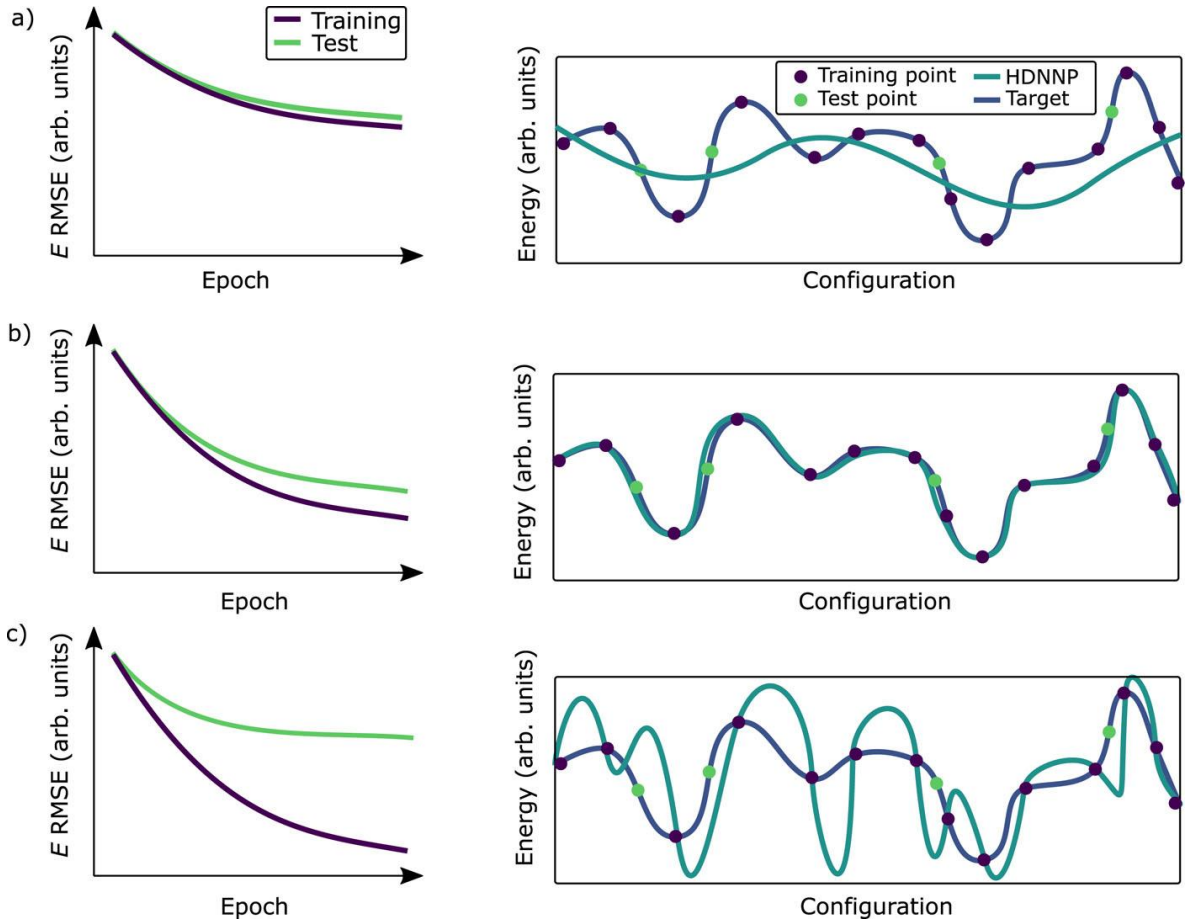
- Typical datasets contain about 10 000 structures (with roughly 100 atoms each), i.e., 10 000 total energies and about 1 000 000 force vectors for each atom.



A. M. Tokita et al., J. Chem. Phys. 159, 121501 (2023).

3.5.2 Example 3: Behler-Parrinello Neural Network Potentials (BP)

Underfitting and overfitting



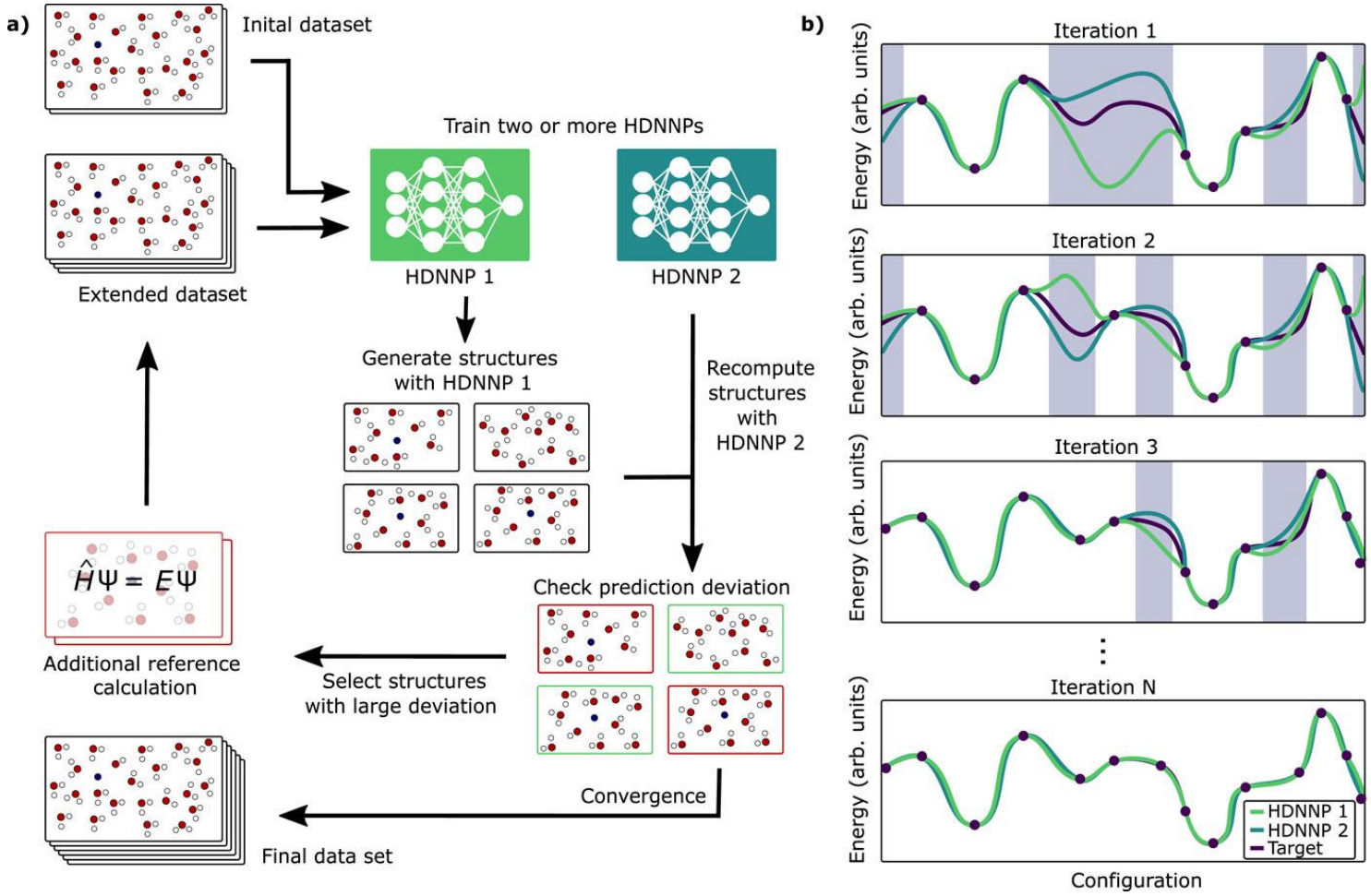
Underfitting

Overfitting

A. M. Tokita et al., J. Chem. Phys. 159, 121501 (2023).

3.5.2 Example 3: Behler-Parrinello Neural Network Potentials (BP)

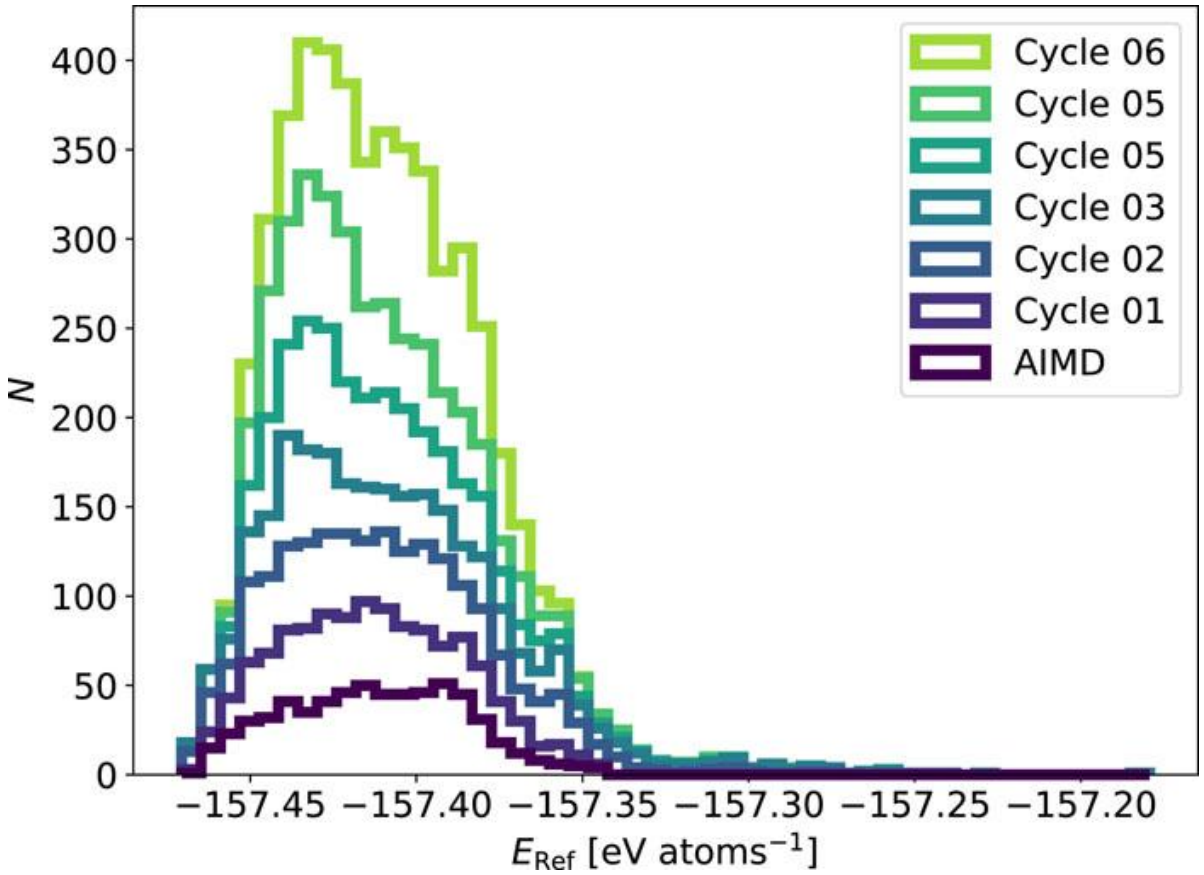
Active learning

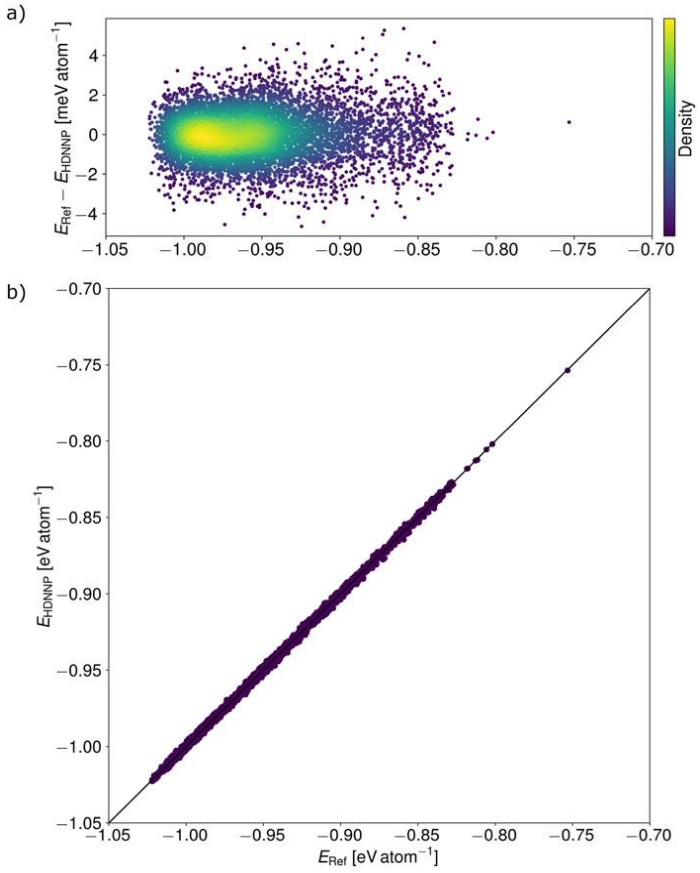


A. M. Tokita et al., J. Chem. Phys. 159, 121501 (2023).

3.5.2 Example 3: Behler-Parrinello Neural Network Potentials (BP)

Simple application: Lithium hydroxide





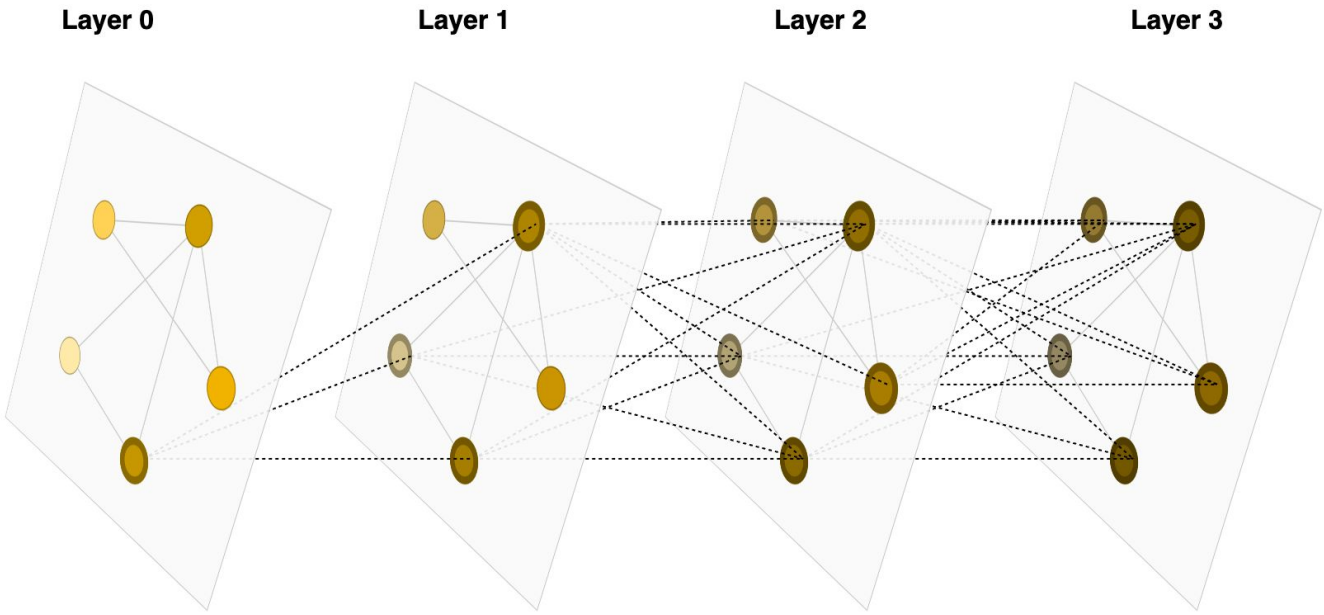
A. M. Tokita et al., J. Chem. Phys. 159, 121501 (2023).

3.5.3 Example 4: Deep Learning Potentials

Convolutional graph neural networks (message passing neural networks)

Introductory material and interactive graphics

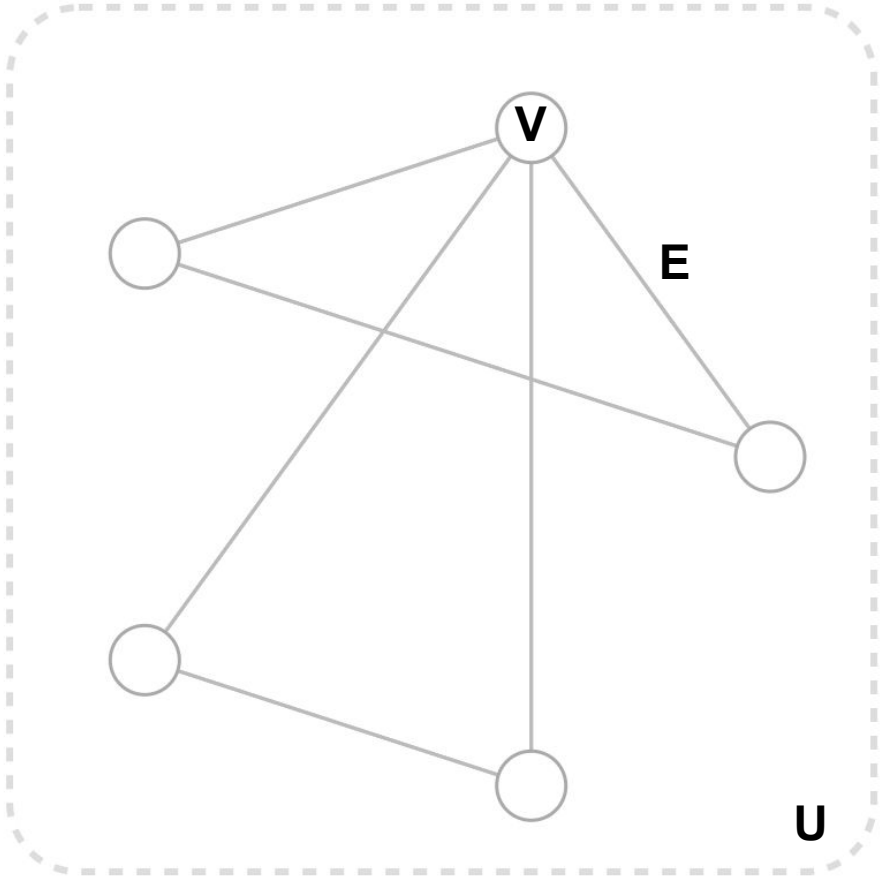
- “A Gentle Introduction to Graph Neural Networks” (doi.org/10.23915/distill.00033)
- “Understanding Convolutions on Graphs” (doi.org/10.23915/distill.00032)



A Gentle Introduction to Graph Neural Networks, doi.org/10.23915/distill.00033.

3.5.3 Example 4: Deep Learning Potentials

Convolutional graph neural networks (message passing neural networks)



V Vertex (or node) attributes

e.g., node identity, number of neighbors

E Edge (or link) attributes and directions

e.g., edge identity, edge weight

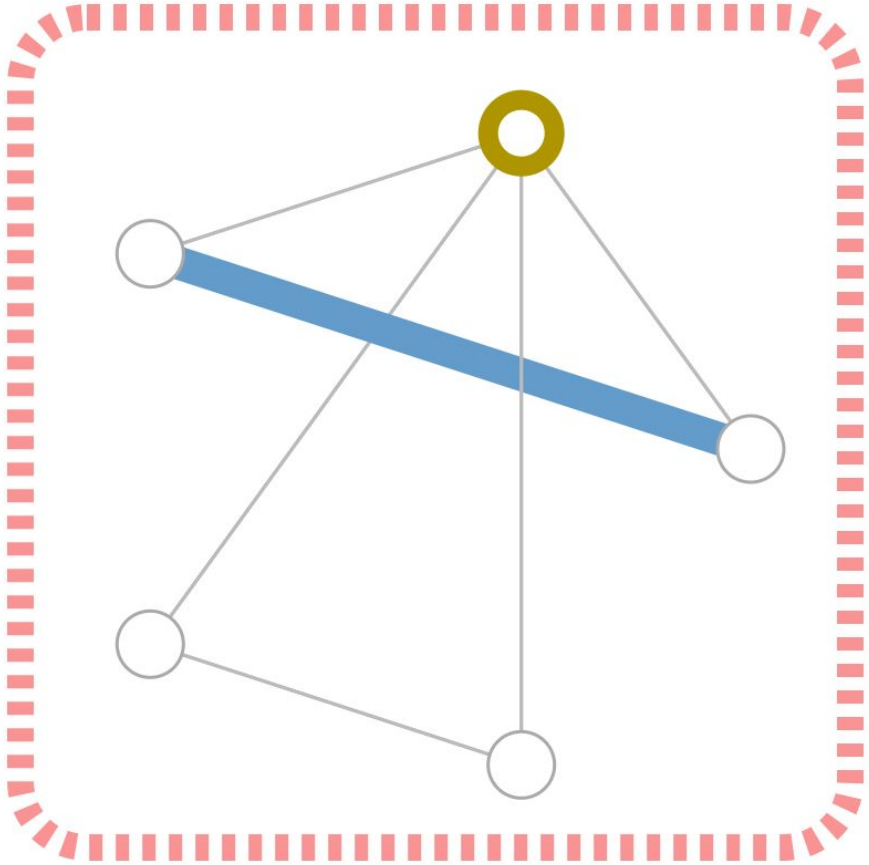
U Global (or master node) attributes

e.g., number of nodes, longest path

A Gentle Introduction to Graph Neural Networks, doi.org/10.23915/distill.00033.

3.5.3 Example 4: Deep Learning Potentials

Convolutional graph neural networks (message passing neural networks)



Vertex (or node) embedding

Edge (or link) attributes and embedding

Global (or master node) embedding

A Gentle Introduction to Graph Neural Networks, doi.org/10.23915/distill.00033.

3.5.3 Example 4: Deep Learning Potentials

Convolutional graph neural networks (message passing neural networks)

Images can represented as graphs

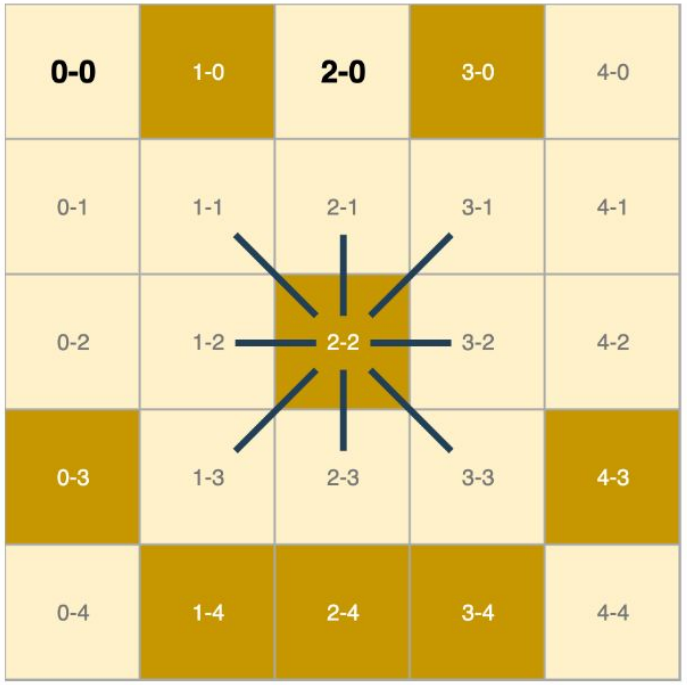
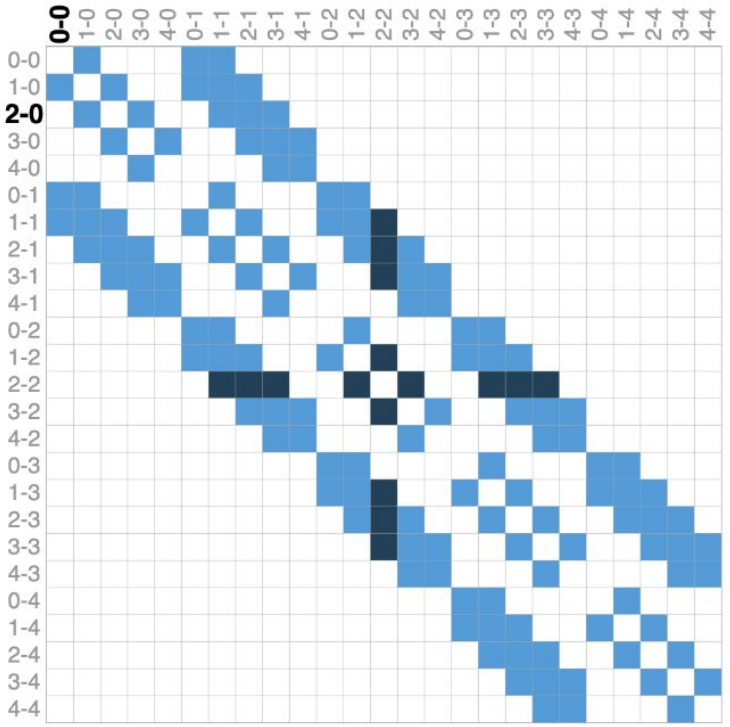
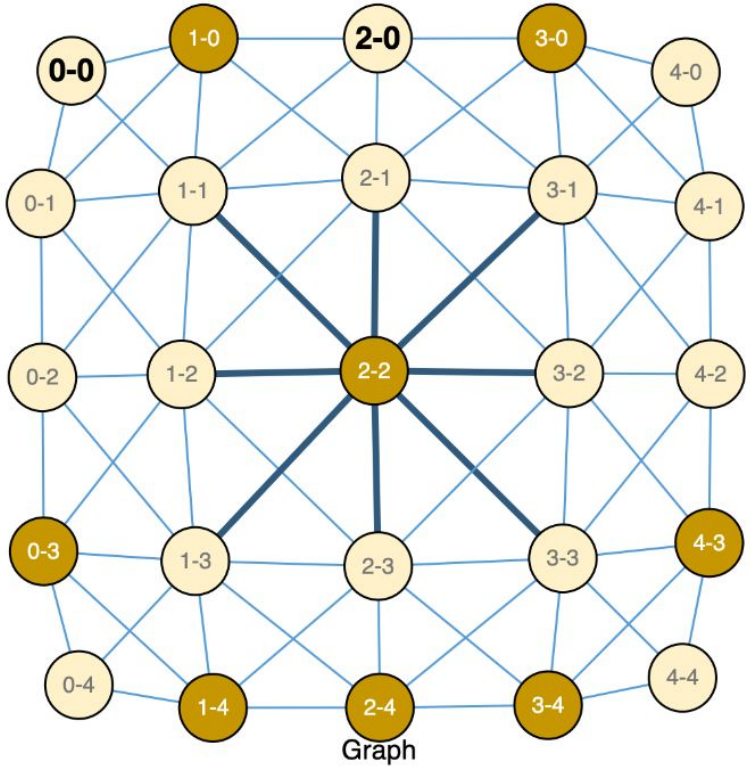


Image Pixels



Adjacency Matrix



Graph

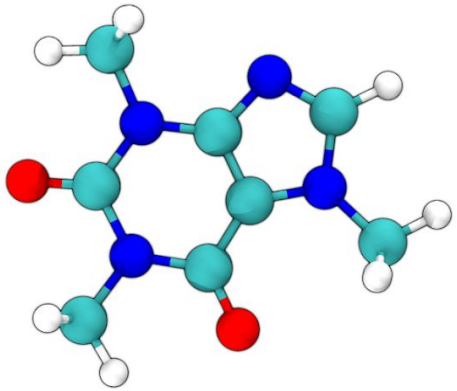
A Gentle Introduction to Graph Neural Networks, doi.org/10.23915/distill.00033.

3.5.3 Example 4: Deep Learning Potentials

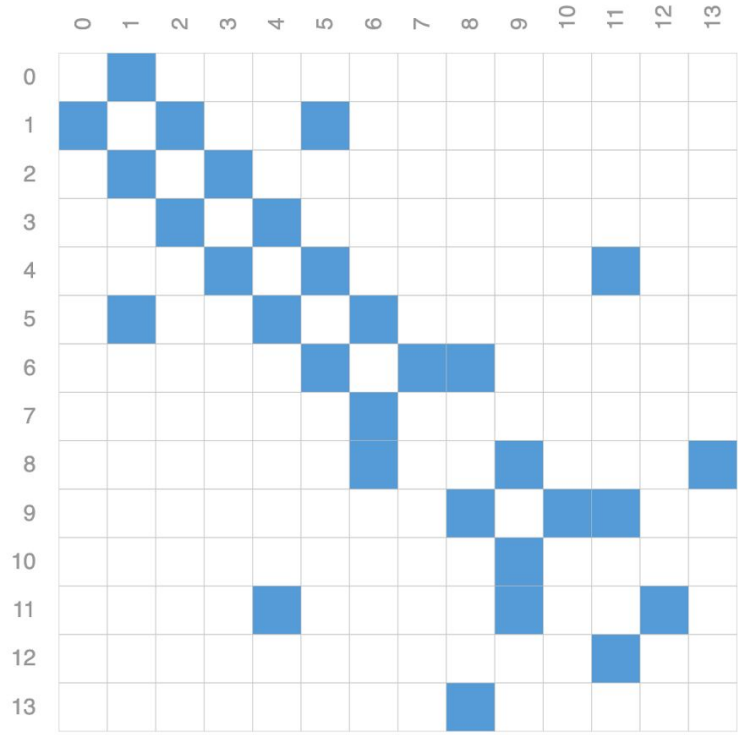
Convolutional graph neural networks (message passing neural networks)

More importantly, also molecules can represented as graphs.

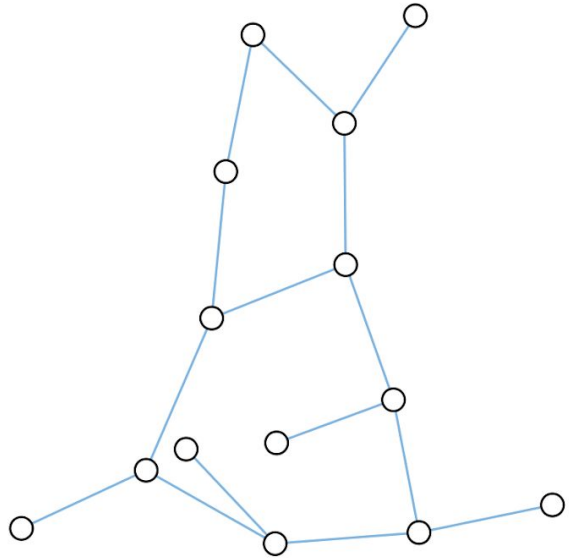
Caffeine molecule



Adjacency matrix



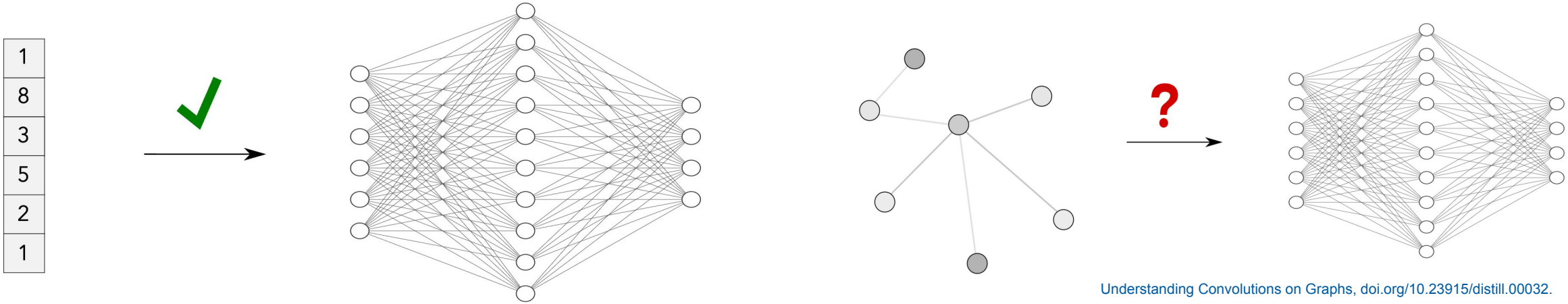
Graph representation



A Gentle Introduction to Graph Neural Networks, doi.org/10.23915/distill.00033.

3.5.3 Example 4: Deep Learning Potentials

Convolutional graph neural networks (message passing neural networks)



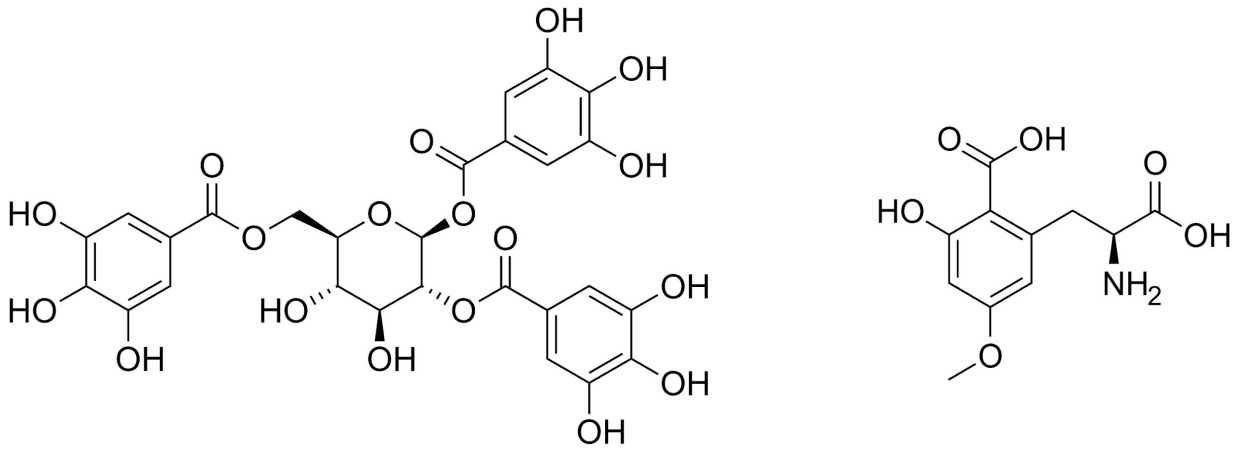
Understanding Convolutions on Graphs, doi.org/10.23915/distill.00032.

3.5.3 Example 4: Deep Learning Potentials

Convolutional graph neural networks (message passing neural networks)

Challenges with graphs: Lack of consistency

- Graphs are very flexible mathematical models, but lack consistent structure across instances. (toxic vs. non-toxic molecules)
- Molecules may have different numbers of atoms.
- Atoms may be of different types.
- Each atom may have different numbers of connections.
- Connections can have different strengths (chemical bonds).



Left: A non-toxic 1,2,6-trigalloyl-glucose molecule.

Right: A toxic caramboxin molecule.

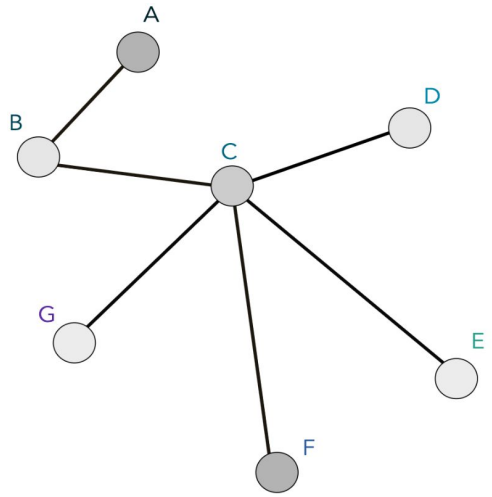
Understanding Convolutions on Graphs, doi.org/10.23915/distill.00032.

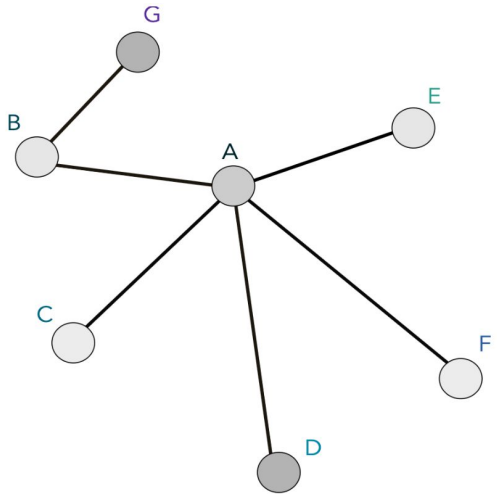
3.5.3 Example 4: Deep Learning Potentials

Convolutional graph neural networks (message passing neural networks)

Challenges with graphs: Node-order equivariance

- Graphs often have no inherent ordering. (molecules)
- But representing graphs as vectors requires us to use a fixed order. (See example on the left.)
- Want algorithms to be node-order equivariant!





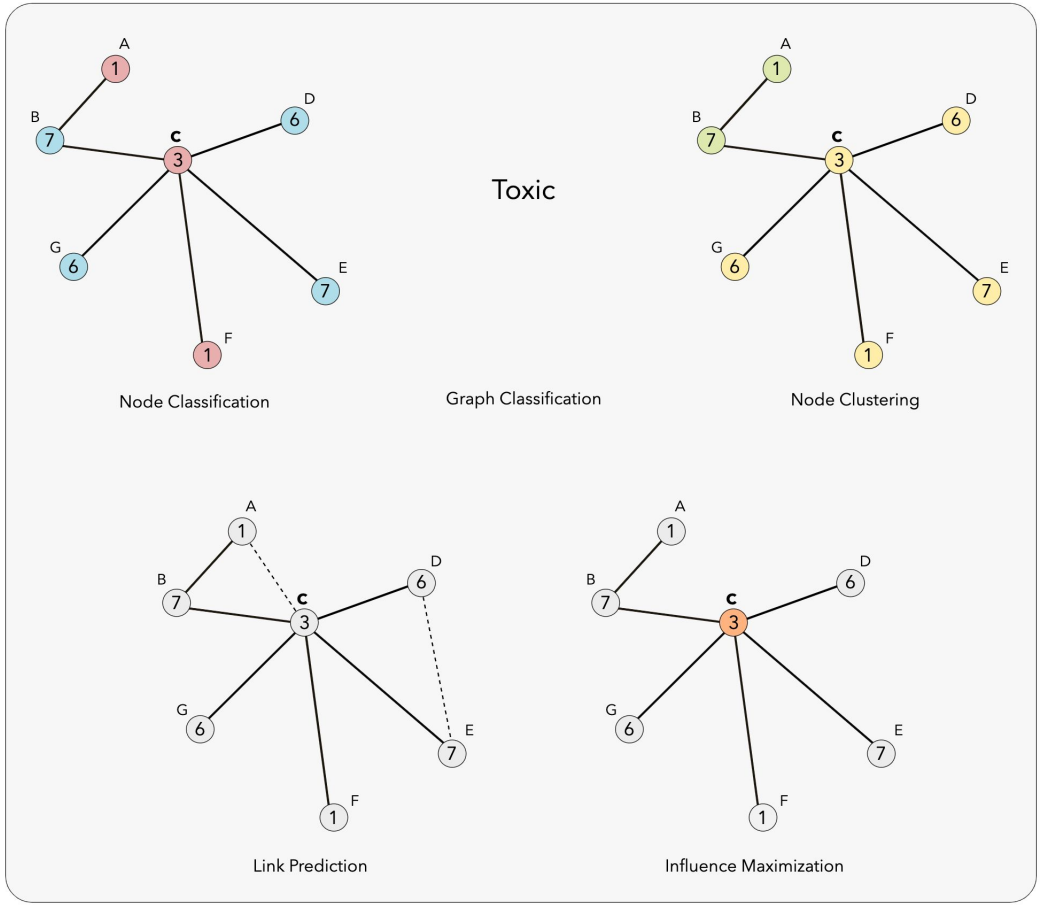
Understanding Convolutions on Graphs, doi.org/10.23915/distill.00032.

3.5.3 Example 4: Deep Learning Potentials

Convolutional graph neural networks (message passing neural networks)

Common problems for graphs

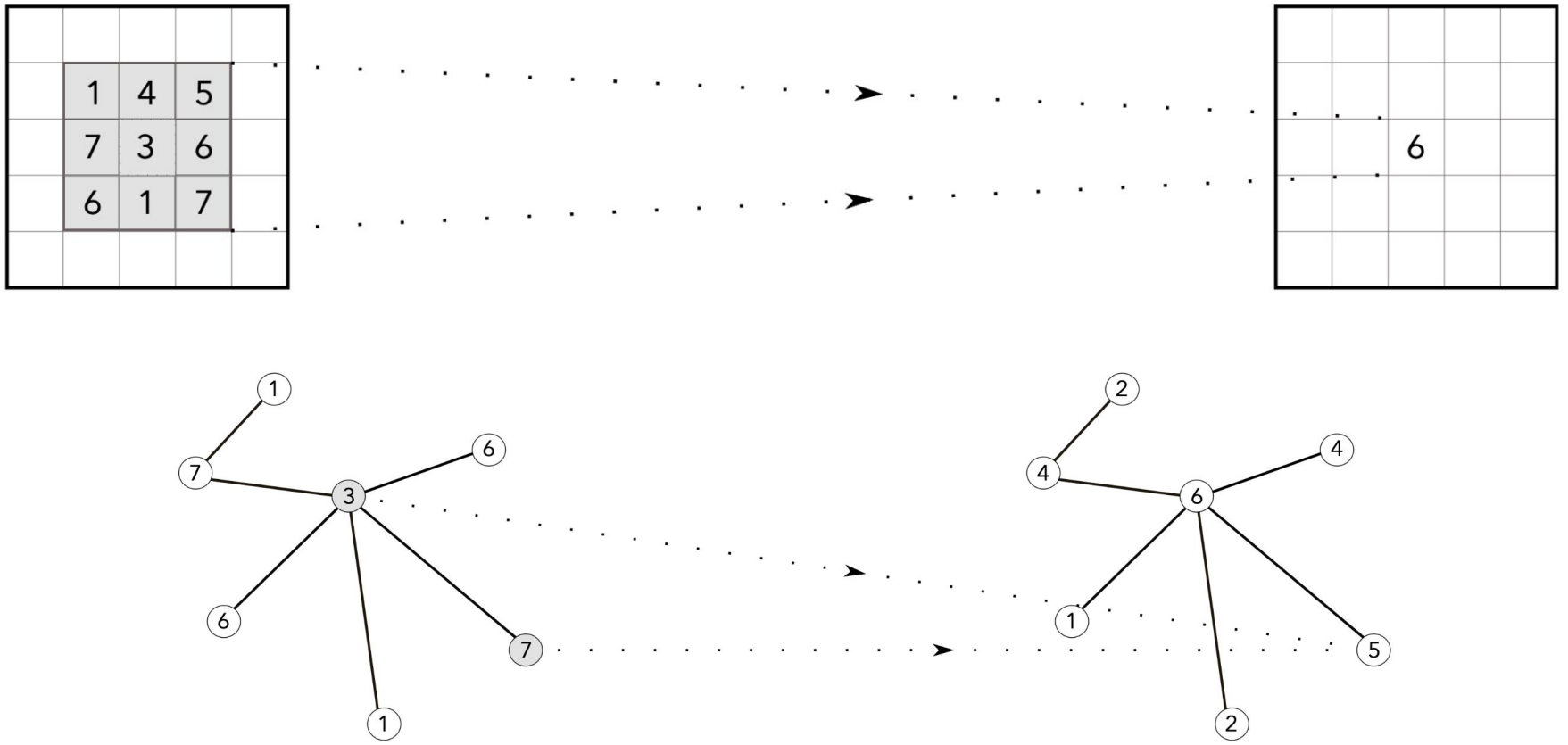
- **Node Classification:** Classifying individual nodes.
- **Graph Classification:** Classifying entire graphs.
- **Node Clustering:** Grouping together similar nodes based on connectivity.
- **Link Prediction:** Predicting missing links.
- **Influence Maximization:** Identifying influential nodes.



Understanding Convolutions on Graphs, doi.org/10.23915/distill.00032.

3.5.3 Example 4: Deep Learning Potentials

Convolutional graph neural networks (message passing neural networks)



Localized Convolution in GNNs

Understanding Convolutions on Graphs, doi.org/10.23915/distill.00032.

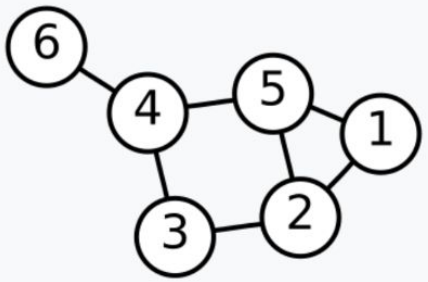
3.5.3 Example 4: Deep Learning Potentials

Convolutional graph neural networks (message passing neural networks)

Polynomial filters on graphs

Consider a useful concept — the graph Laplacian:

$$D \quad - \quad A \quad = \quad L$$

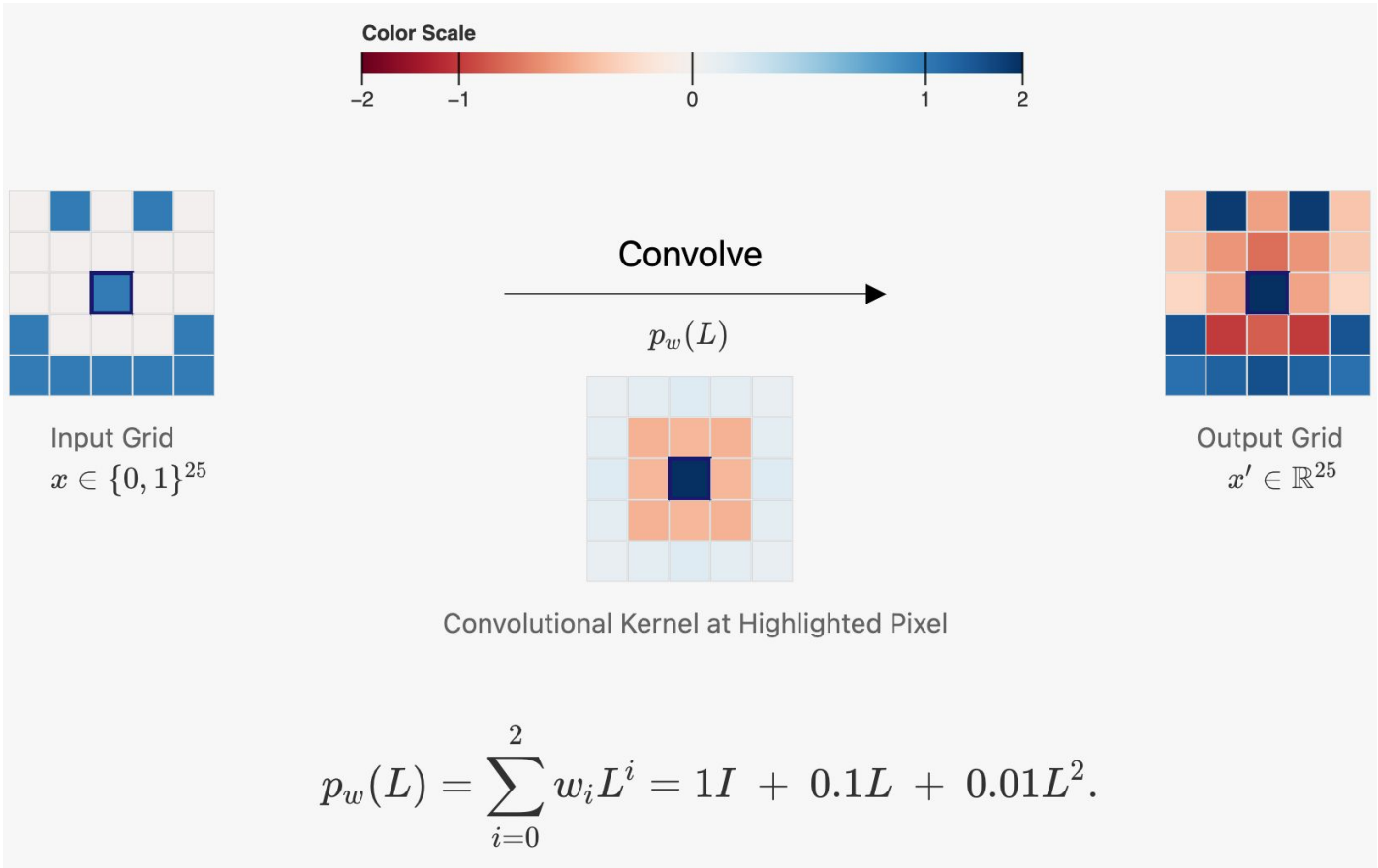
Labelled graph	Degree matrix	Adjacency matrix	Laplacian matrix
	$\begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 2 & -1 & 0 & 0 & -1 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & -1 & 0 & -1 & 3 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{pmatrix}$

3.5.3 Example 4: Deep Learning Potentials

Convolutional graph neural networks (message passing neural networks)

Polynomial filters on graphs

- Build polynomials based on graph Laplacian which are equivalent of filters.
- Convolution at given node occurs only with nodes not more than a certain number of hops away.
- Polynomial filters are thus localized.
- Polynomial filters are node-order equivariant.



$$p_w(L) = \sum_{i=0}^{2} w_i L^i = 1I + 0.1L + 0.01L^2.$$

Understanding Convolutions on Graphs, doi.org/10.23915/distill.00032.

3.5.3 Example 4: Deep Learning Potentials

Convolutional graph neural networks (message passing neural networks)

Node-order equivariant polynomial filters

Start with the original features.

$$h^{(0)} = x$$

Then iterate, for $k = 1, 2, \ldots$ upto K:

$$p^{(k)} = p_{w^{(k)}}(L)$$

Compute the matrix $p^{(k)}$ as the polynomial defined by the filter weights $w^{(k)}$ evaluated at L.

$$g^{(k)} = p^{(k)} \times h^{(k-1)}$$

Multiply $p^{(k)}$ with $h^{(k-1)}$: a standard matrix-vector multiply operation.

$$h^{(k)} = \sigma\left(g^{(k)}\right)$$

Apply a non-linearity σ to $g^{(k)}$ to get $h^{(k)}$.

Color Codes:

- Computed node embeddings.
- Learnable parameters.

Understanding Convolutions on Graphs, doi.org/10.23915/distill.00032.

3.5.3 Example 4: Deep Learning Potentials

Convolutional graph neural networks (message passing neural networks)

Computing node-order equivariant embeddings

$$h_v^{(0)} = x_v \quad \text{for all } v \in V.$$

Node v's initial embedding. ... is just node v's original features.

and for $k = 1, 2, \ldots$ upto K:

$$h_v^{(k)} = f^{(k)}\left(W^{(k)} \cdot \frac{\sum_{u \in \mathcal{N}(v)} h_u^{(k-1)}}{|\mathcal{N}(v)|} + B^{(k)} \cdot h_v^{(k-1)} \right) \quad \text{for all } v \in V.$$

Node v's embedding at step k. Mean of v's neighbour's embeddings at step $k-1$. Node v's embedding at step $k-1$.

Color Codes:

- Embedding of node v.
- Embedding of a neighbour of node v.
- (Potentially) Learnable parameters.

Predictions can be made at each node by using the final computed embedding:

$$\hat{y}_v = \text{PREDICT}(h_v^{(K)})$$

where PREDICT is generally another neural network, learnt together with the GCN model.

For each step k, the function $f^{(k)}$, matrices $W^{(k)}$ and $B^{(k)}$ are shared across all nodes.

This allows the GCN model to scale well, because the number of parameters in the model is not tied to the size of the graph.

- **Key idea:** Consider different kinds of aggregation and combination steps, beyond what is possible using polynomial filters.
- These can be viewed as “message-passing” between adjacent nodes: after each step, every node receives some information from its neighbours.
- By iteratively repeating the 1-hop localized convolutions K times, the receptive field of the convolution effectively includes all nodes up to K hops away.

Understanding Convolutions on Graphs, doi.org/10.23915/distill.00032.

3.5.4 Example 5: MACE potential

- Represent the state of each node in a given layer by a tuple:

$$\sigma_i^{(t)} = (\boldsymbol{r}_i, z_i, \boldsymbol{h}_i^{(t)})$$

charge of atoms; position of atoms; learnable features

- A message is created for each node by pooling over its neighbors:

$$\boldsymbol{m}_i^{(t)} = \bigoplus_{j \in \mathcal{N}(i)} M_t(\sigma_i^{(t)}, \sigma_j^{(t)})$$

learnable message function

- In the update step, the message is transformed into new features:

$$\boldsymbol{h}_i^{(t+1)} = U_t(\sigma_i^{(t)}, \boldsymbol{m}_i^{(t)})$$

learnable update function

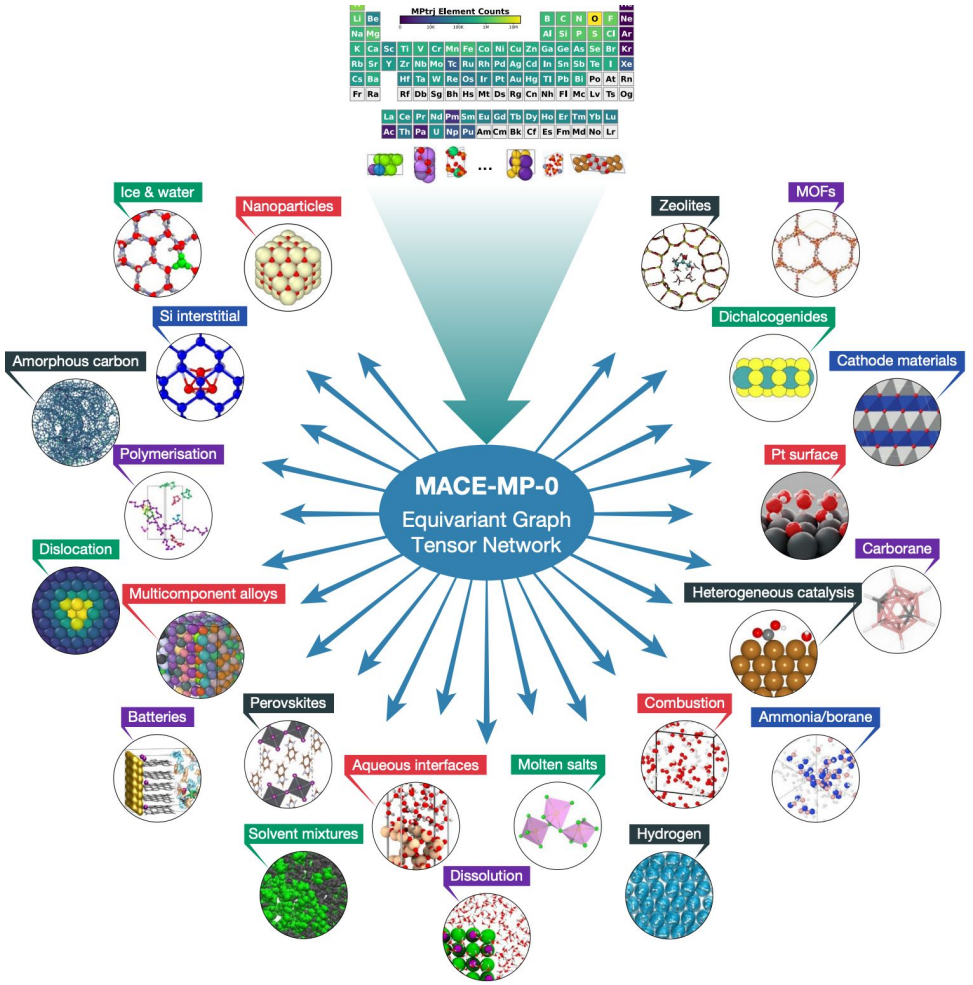
- After a given number of message construction and update steps, the learnable readout functions map the node states to the target — the local energy of a given atom:

$$E_i = \sum_{t=1}^{T} \mathcal{R}_t(\sigma_i^{(t)})$$

learnable readout function

3.5.4 Example 5: Advanced Potentials

Foundation models



I. Batatia, et al., arXiv:2401.00096 (2023).