

GNN for Materials Design

Atomistic structure, Spectral, Image and Text data

Kamal Choudhary
<https://jarvis.nist.gov/>
NIST, Gaithersburg, MD, USA
APS 2023



Joint Automated Repository for Various Integrated Simulations

Outline

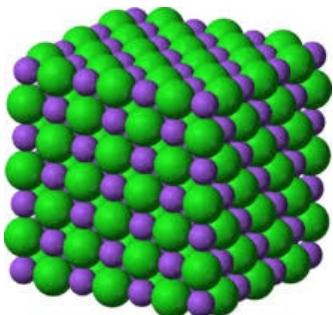
Contents

- Motivation
- Introduction
- ALIGNN
- AtomVision
- ChemNLP
- Leaderboard
- Summary

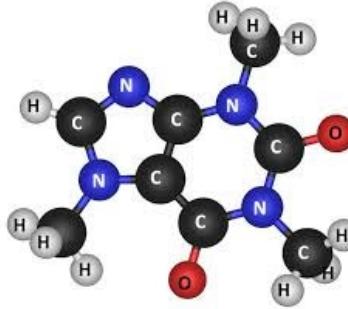
Motivation

- Accelerate traditional computational and experimental methods
- Automate experimental data analysis,
- Discover new materials,
- Develop new methods,
- Find new physical equations/phenomenon,
- Enhance collaboration,
- Uncertainty quantification, reproducibility,...

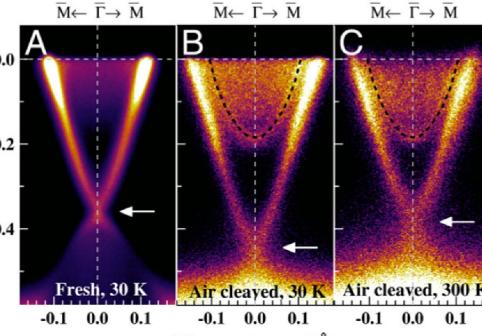
Modularity: discrete, continuous, images, text



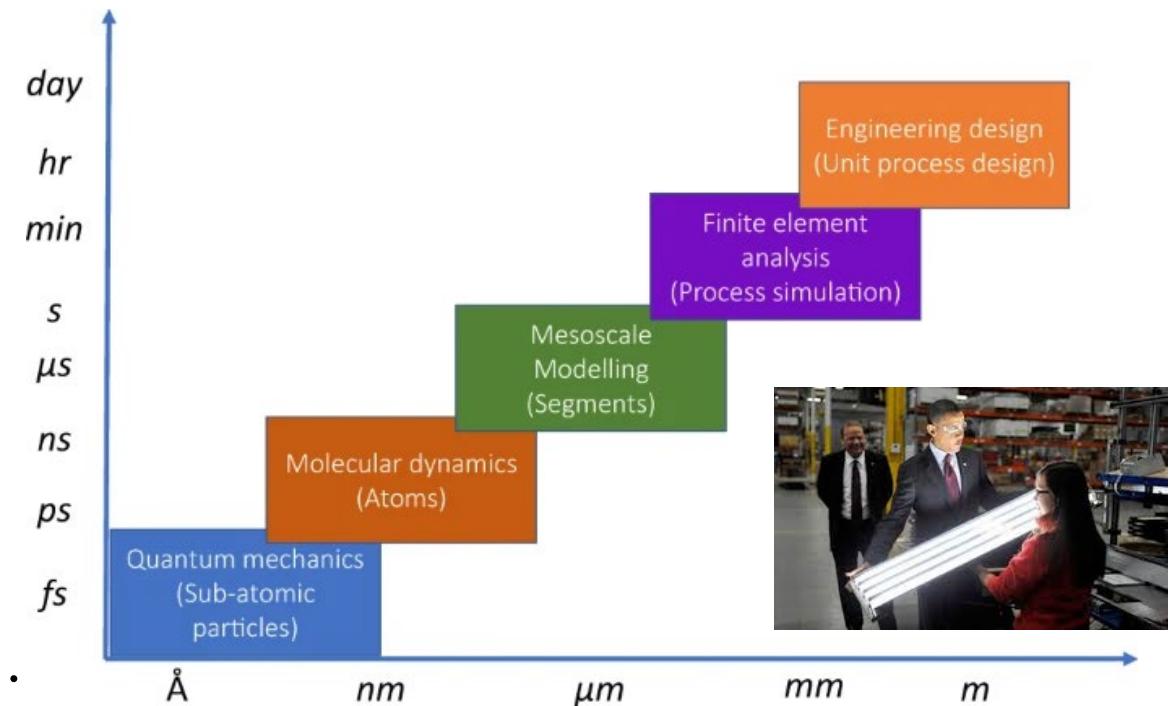
Crystals



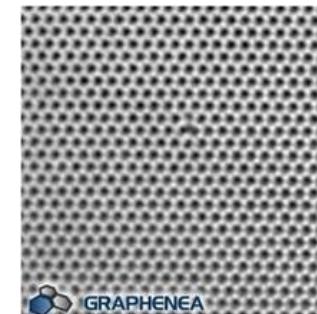
Molecules



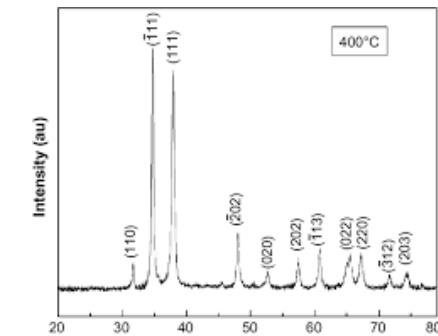
Band-structure



Materials Genome Initiative 2011



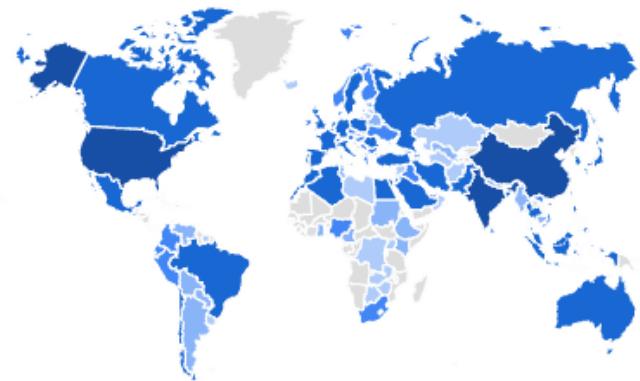
TEM images



XRD spectra

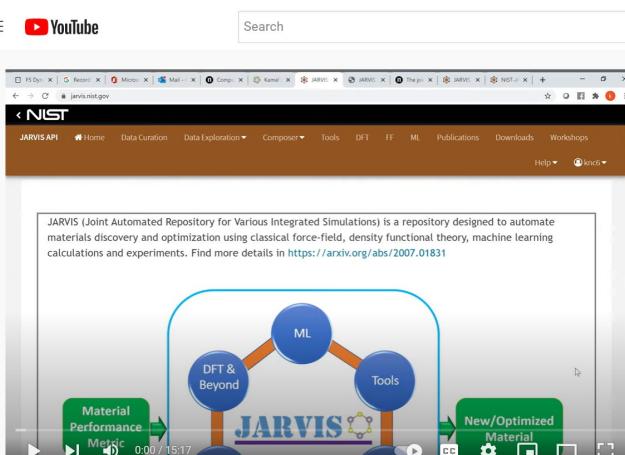
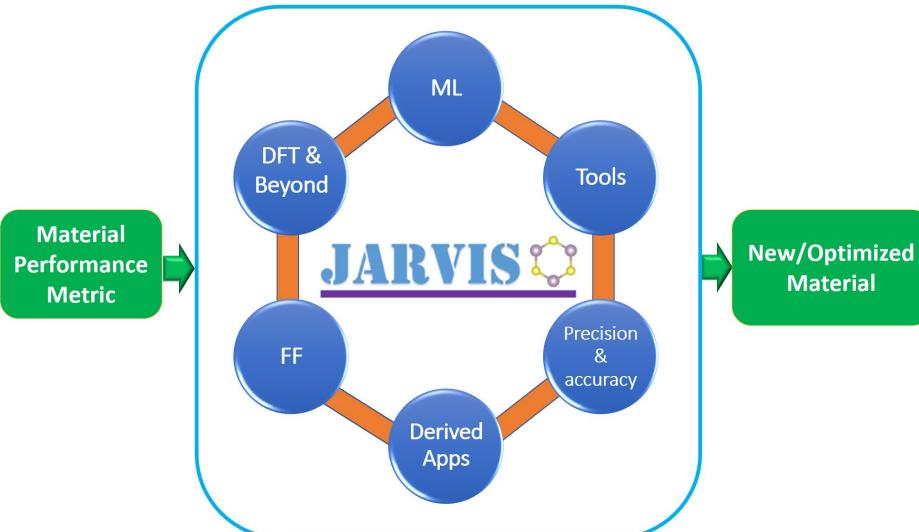


Databases, Tools, Events, Outreach



<https://jarvis.nist.gov>

Requires login credentials, free registration



Established: January 2017

Published: >40 articles

Users: >15000+ users worldwide

Materials: >70000, millions of properties

Events:

- Quantum Matters in Materials Science (QMMS)
- Artificial Intelligence for Materials Science (AIMS)
- JARVIS-School

User-comments:

- “There are many different theoretical levels on which you can approach the field. JARVIS is unusual in that it spans more levels than other databases.”
- “A pure gold-mine for the data-quality effort...”
- “Thanks for your generous sharing. Your works inspire me a lot.”
- “You guys are doing something really beneficial...”
- “I find JARVIS-DFT very useful for my research...”



Tools

 usnistgov / **jarvis** Public

[Code](#) [Issues](#) 32 [Pull requests](#) 4

[Disc](#)

 usnistgov / **alignn** Public

[Code](#) [Issues](#) 16 [Pull requests](#)

[Actions](#)

 usnistgov / **chemnlp** Public

[Code](#) [Issues](#) [Pull requests](#)

[Actions](#)

[Projects](#)

 usnistgov / **atomvision** Public

generated from usnistgov/opensource-repo

[Code](#) [Issues](#) 1 [Pull requests](#) 1

[Actions](#)

[Projects](#)

 usnistgov / **atomqc** Public

generated from usnistgov/opensource-repo

[Code](#) [Issues](#) 1 [Pull requests](#)

[Actions](#)

[Projects](#)

 JARVIS-Materials-Design / **jarvis-tools-notebooks**

[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#)

 usnistgov / **tb3py** Public

generated from usnistgov/opensource-repo

[Code](#) [Issues](#) 2 [Pull requests](#)

[Actions](#)

[Projects](#)

 deepmaterials / **dlmatreview** Public

[Code](#) [Issues](#) 3 [Pull requests](#) [Actions](#)

Introduction: deep learning for materials

Artificial Intelligence, AI
(Generic term, mimic cognitive functions)

Machine Learning, ML
(Number of samples > 100)

Linear Regression, Random Forest, Decision Trees, Gaussian Processes, ...

Deep Learning, DL
(Number of samples > 500)

Artificial Neural Network (ANN), Convolution Neural Network (CNN), Graph Neural Network (GNN), Variational Encoders (VAE), Generative Adversarial Network (GAN), Recurrent Neural Network (RNN), Deep Reinforcement Learning (DRL), ...

Chemical Formula,
SMILES, Fragments

Atomic Structure
(Molecules, Solids,
Proteins)

Text/Literature

XRD, XAS, Raman, NMR,
UV-vis, XANES,
Electron/Phonon DOS

SEM, STM, STEM images

npj | computational materials

Explore content ▾ About the journal ▾ Publish with us ▾

[nature](#) > [npj computational materials](#) > [review articles](#) > [article](#)

Review Article | [Open Access](#) | Published: 05 April 2022

Recent advances and applications of deep learning methods in materials science

[Kamal Choudhary](#)✉, [Brian DeCost](#), [Chi Chen](#), [Anubhav Jain](#), [Francesca Tavazza](#), [Ryan Cohn](#), [Cheol Woo Park](#), [Alok Choudhary](#), [Ankit Agrawal](#), [Simon J. L. Billinge](#), [Elizabeth Holm](#), [Shyue Ping Ong](#) & [Chris Wolverton](#)

[npj Computational Materials](#) **8**, Article number: 59 (2022) | [Cite this article](#)

ALIGNN

Atomistic Line Graph Neural Network
<https://github.com/usnistgov/alignn>

usnistgov / alignn Public

Notifications Fork 50 Star 102

Code Issues 18 Pull requests 2 Actions Projects Security Insights

main 22 branches 30 tags Go to file Code

knc6 Merge pull request #87 from usnistgov/develop ... ✓ 736ec73 on Jan 10 450 commits

.github	FIX IGNITE.	3 months ago
alignn	Add package_data in setup.py for loading config and model ...	last month
.gitignore	Modified gitignore	6 months ago
.pre-commit-config.yaml	reduce range for momentum annealing	2 years ago
LICENSE.rst	Update and rename LICENSE to LICENSE.rst	2 years ago
README.md	Added supercon model, added FF dataset for elements from...	last month
pyproject.toml	fix newline in pyproject.toml	2 years ago
setup.py	Add package data in setup.py for loading config and model ...	last month

About Atomistic Line Graph Neural Network
jarvis.nist.gov/jalignn/

Readme View license 102 stars 10 watching 50 forks

Releases 30 v2023.01.10 Latest on Jan 10

Notebook

The screenshot shows a GitHub notebook page. At the top, there's a URL bar with the address: github.com/knc6/jarvis-tools-notebooks/blob/master/jarvis-tools-notebooks/Training_ALIGNN_model_example.ipynb. Below the URL bar, there are two tabs: "Code" and "Blame". The "Code" tab is selected, showing statistics: 2568 lines (2568 loc) · 128 KB. To the right of the stats, there's a repository link: [JARVIS-Materials-Design / jarvis-tools-notebooks](#). Further down, there's a "Open in Colab" button. At the bottom of the page, there's a navigation bar with links: Code, Issues, Pull requests, Actions, and Projects. The "Code" link is highlighted with a red underline.

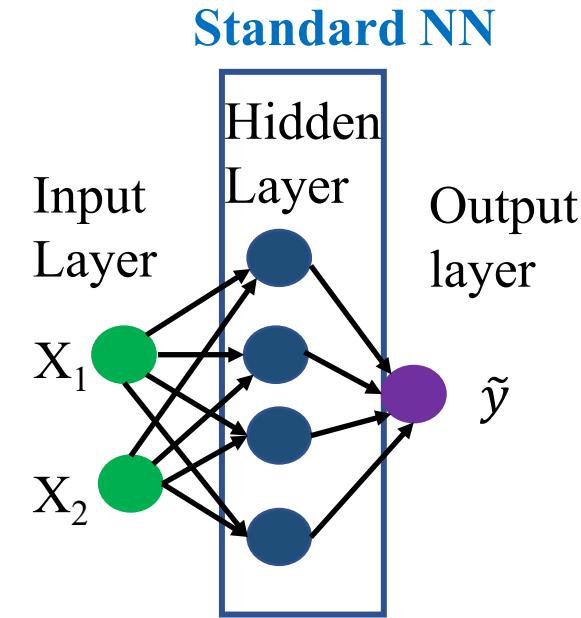
Table of contents

1. Installing [ALIGNN](#)
2. Example training for regression on 50 materials,
3. Using pre-trained models to make fast predictions
4. Using ALIGNN-FF model to predict the unrelaxed energy (fast), optimized structure and energy, and EV curve
5. Train ALIGNN-FF on a new dataset
6. Training [JARVIS-DFT](#) 2D exfoliation energy model
7. Training [QM9](#) U0 model

In [1]:

```
!pip install alignn
```

From ANNs to Graph Convolution Networks



1) Forward propagation

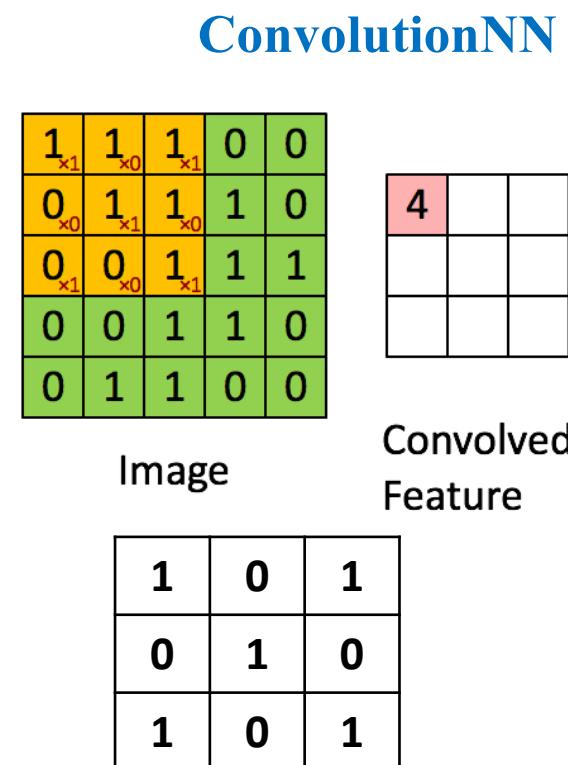
$$z^{[l]} = W^{[l]} a^{[l-1]} + b^{[l]}$$

$$a^{[1]} = \sigma(z^{[l]}); \quad a^{[0]} = X$$

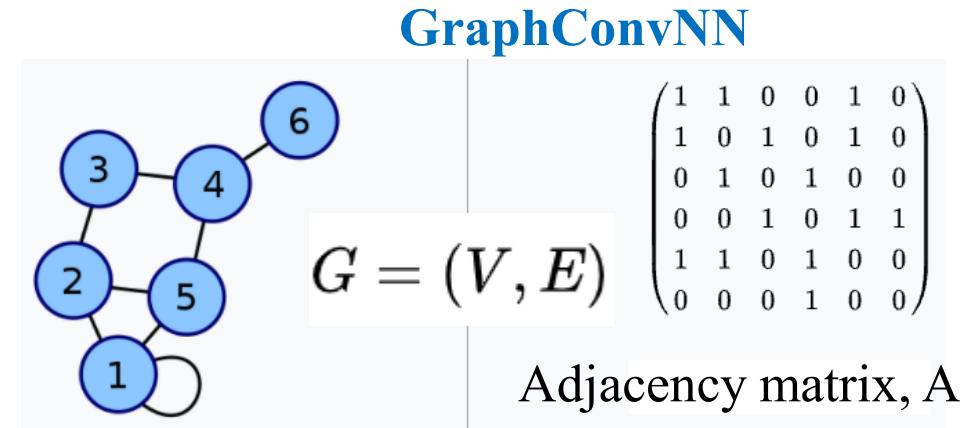
2) Cost, $J(W, b) = f(y - \tilde{y})$

3) Gradient descent (∇J):
minimize cost with W, b

4) Backpropagation:
chain rule to get, $\frac{\partial J}{\partial W}$



- 1) Convolution:
element-wise multiplication & sum
- 2) Pool: Max, Average, Sum
- 3) Fully Connected: Standard NN
Shared weights (Learnable filters),
regularized version of NNs



Types: un/weighted, un/directed, line,
Hetero/Homogenous, Multigraph

- 1) Adjacency matrix, $N \times N$ (N : #nodes),
- 2) D: degree of node
- 3) Update node representation using message passing, GPU efficient
- 4) Update equation is local, neighborhood of a node only, independent of graph size

$$\hat{A} = \widetilde{D}^{-\frac{1}{2}} A \widetilde{D}^{-\frac{1}{2}} \quad H^{l+1} = \sigma(W \hat{A} H^l)$$

$$h_i^{\ell+1} = f(h_i^\ell, \{h_j^\ell\}_{j \in \mathcal{N}_i})$$

Computer Science > Machine Learning

[Submitted on 2 Mar 2020 (v1), last revised 3 Jul 2020 (this version, v3)]

Benchmarking Graph Neural Networks

Vijay Prakash Dwivedi, Chaitanya K. Joshi, Thomas Laurent, Yoshua Bengio, Xavier Bresson

Graph neural networks (GNNs) have become the standard toolkit for analyzing and learning from data on graphs. As the field grows, it becomes critical to identify key architectures and validate new ideas that generalize to larger, more complex datasets. Unfortunately, it has been increasingly difficult to gauge the effectiveness of new models in the absence of a standardized benchmark with consistent experimental settings. In this paper, we introduce a reproducible GNN benchmarking framework, with the facility for researchers to add new models conveniently for arbitrary datasets. We demonstrate the usefulness of our framework by presenting a principled investigation into the recent Weisfeiler-Lehman GNNs (WL-GNNs) compared to message passing-based graph convolutional networks (GCNs) for a variety of graph tasks, i.e. graph regression/classification and node/link prediction, with medium-scale datasets.

Comments: Benchmarking framework on GitHub at [this https URL](https://github.com/graphdeeplearning/benchmarking-gnns)Subjects: **Machine Learning (cs.LG)**; Machine Learning (stat.ML)

Cite as: arXiv:2003.00982 [cs.LG]

(or arXiv:2003.00982v3 [cs.LG] for this version)

[graphdeeplearning / benchmarking-gnns](https://github.com/graphdeeplearning/benchmarking-gnns) Public

[Watch](#) 51 [Star](#) 1.5k [Fork](#) 275

[Code](#) [Issues 3](#) [Pull requests](#) [Discussions](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#)

[master](#) [6 branches](#) [0 tags](#)

	Go to file	Add file	Code
vijaydwivedi75 Update leaderboard	62c32cc on Jul 28	36 commits	
configs LapPE (#32)		15 months ago	
data minor code clean		11 months ago	
docs Update leaderboard		2 months ago	
layers inbuilt reduce funcs (#55)		10 months ago	

About

Repository for benchmarking graph neural networks

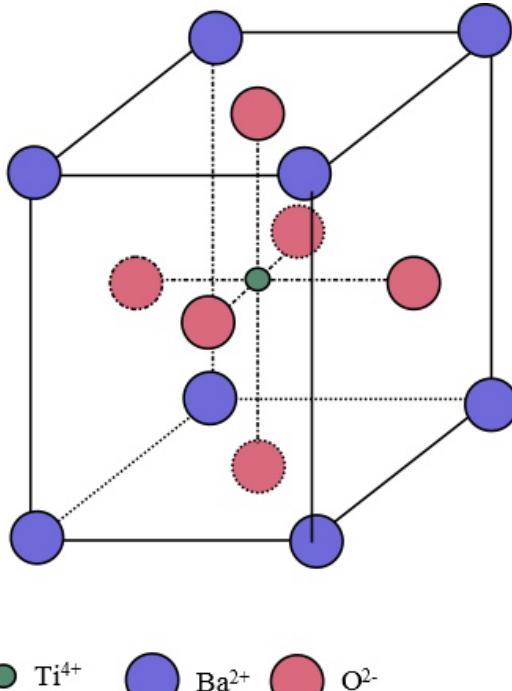
[arxiv.org/abs/2003.00982](#)

[deep-learning](#) [pytorch](#)
[benchmark-framework](#)
[graph-neural-networks](#)
[graph-representation-learning](#) [dgl](#)

Graph Convolution Networks for Materials Scientists

Consider BaTiO₃: start with n_{input} -dimensional atom feature vectors

$$h_{Ti} \ h_{Ba} \ h_O$$



The local environment of Ti can be modeled as:

$$h'_{Ti} = 6f(h_{Ti}, h_O, r_{TiO}) + 8f(h_{Ti}, h_{Ba}, r_{TiBa})$$

In GCNs, f is parameterized by one or more neural network layers

e.g. *edge-gated graph convolution*: [arxiv:1711.07553](https://arxiv.org/abs/1711.07553)

$$f(h_i, h_j, r_{ij}) = \phi_{gate}([h_i; h_j; \phi_{rbf}(r_{ij})]) \odot \phi_{pair}([h_i; h_j])$$

Graph embedding transform and encode the data structure in high dimensional and non-Euclidean feature space to a low dimensional space

$\odot \equiv$ Elementwise multiplication

$[a; b]$ denotes concatenation

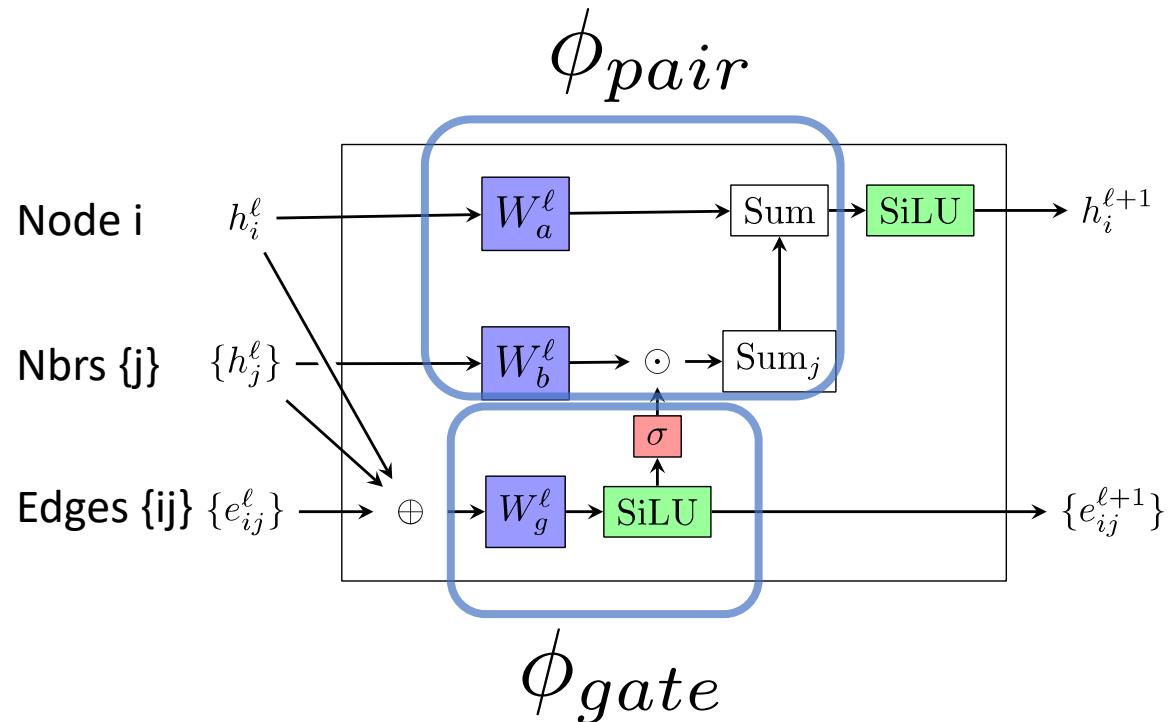
ϕ_{pair} is a neural network modeling pairwise atom interactions

ϕ_{gate} is a neural network modeling the importance of ϕ_{pair} output channels as a function of bond character

GCNs implicitly represent multi-body interactions by composing multiple layers

Edge-gated Graph Convolution Layer

$$\phi_{gate}([h_i; h_j; \phi_{rbf}(r_{ij})]) \odot \phi_{pair}([h_i; h_j])$$



For computational efficiency, the ϕ_{pair} parameters are split into two separate matrices
see <https://docs.dgl.ai/guide/message-efficient.html>

adapted from:
[graphdeeplearning / benchmarking-gnns](https://github.com/graphdeeplearning/benchmarking-gnns) Public

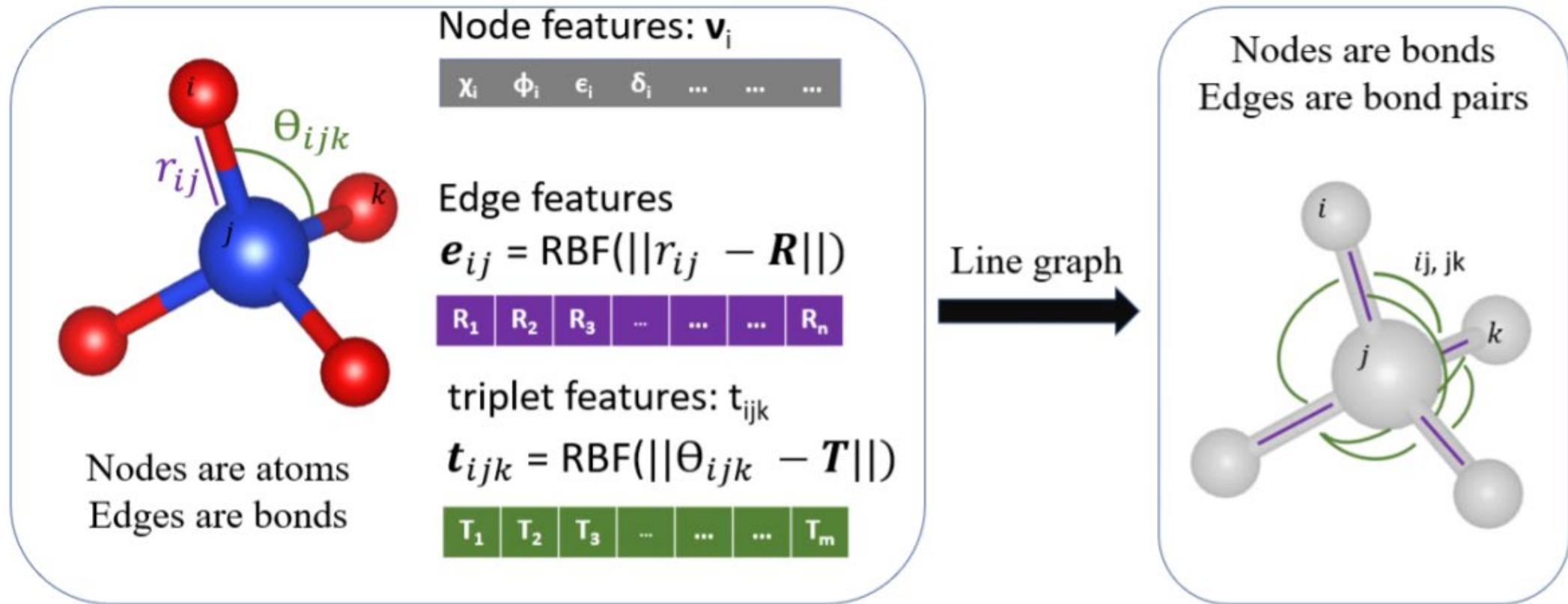
Code Issues Pull requests Discussions

```
1 def forward(
2     self,
3     g: dgl.DGLGraph,
4     node_feats: torch.Tensor,
5     edge_feats: torch.Tensor,
6 ) -> torch.Tensor:
7     """Edge-gated graph convolution.
8     h_i^{l+1} = SiLU(U h_i + sum_{j->i} \eta_{ij} \circ V h_j)
9     """
10    g = g.local_var()
11
12    # compute edge updates, equivalent to:
13    # Softplus(Linear(u || v || e))
14    g.ndata["e_src"] = self.src_gate(node_feats)
15    g.ndata["e_dst"] = self.dst_gate(node_feats)
16    g.apply_edges(fn.u_add_v("e_src", "e_dst", "e_nodes"))
17    m = g.edata.pop("e_nodes") + self.edge_gate(edge_feats)
18
19    g.edata["sigma"] = torch.sigmoid(m)
20    g.ndata["Bh"] = self.dst_update(node_feats)
21    g.update_all(
22        fn.u_mul_e("Bh", "sigma", "m"), fn.sum("m", "sum_sigma_h")
23    )
24    g.update_all(fn.copy_e("sigma", "m"), fn.sum("m", "sum_sigma"))
25    g.ndata["h"] = g.ndata["sum_sigma_h"] / (g.ndata["sum_sigma"] + 1e-6)
26    x = self.src_update(node_feats) + g.ndata.pop("h")
27
28    # node and edge updates
29    x = F.silu(self.bn_nodes(x))
30    y = F.silu(self.bn_edges(m))
```

DGL DEEP GRAPH LIBRARY

Line Graph

Explicitly represent pairwise and triplet (bond angle) interactions using line graph
Possible to extend for n-body, e.g. line graph of line graph



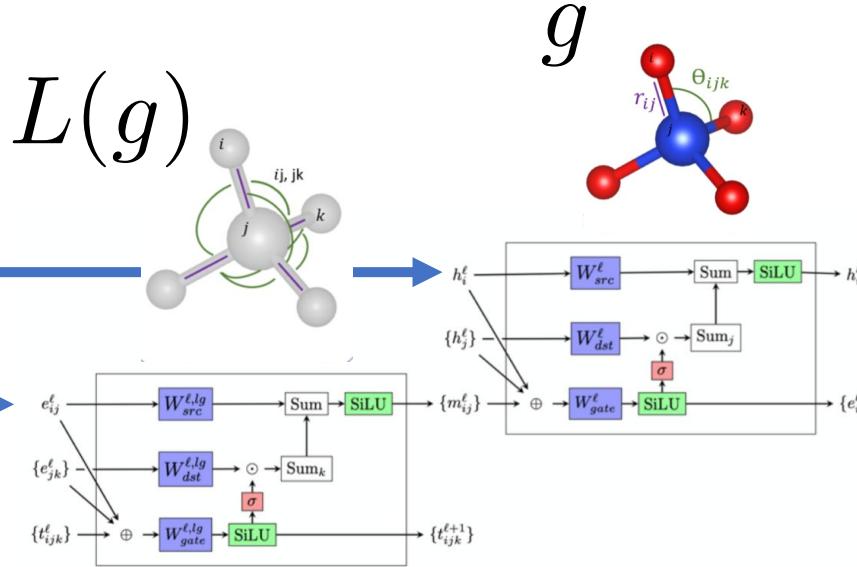
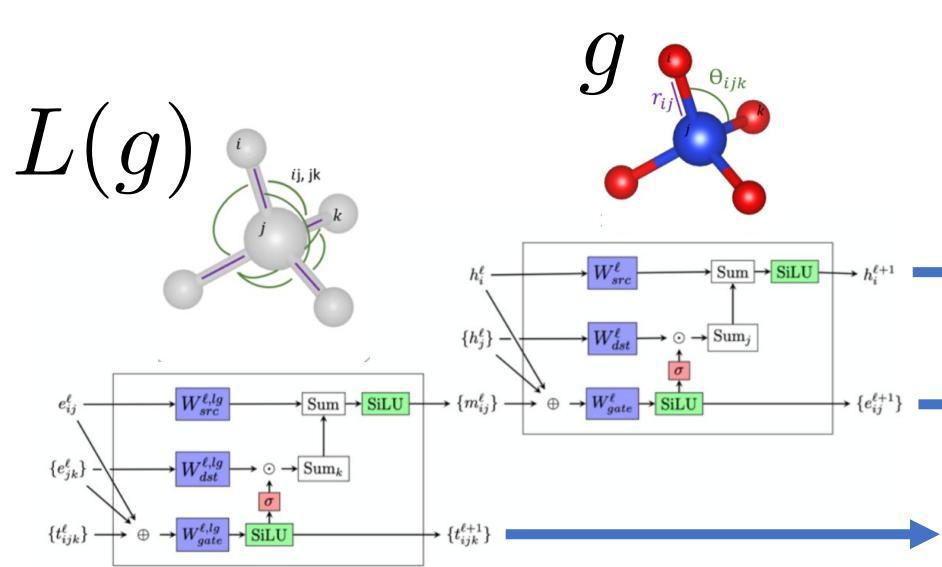
ALIGNN Model

Compose multiple ALIGNN and EdgeGatedGCN layers: learnable local atom environment representation

Global crystal representation: Feature-wise average across atoms in the crystal

Final property regression model: linear model for regression, logistic regression for classification

Initial atom, bond, angle features



```
@staticmethod
def atom_dgl_multigraph(
    atoms=None,
    neighbor_strategy="k-nearest", # arrow points here
    cutoff=8.0,
    max_neighbors=12,
    atom_features="cgcn",
    max_attempts=3,
    id: Optional[str] = None,
    compute_line_graph: bool = True,
    use_canonize: bool = False,
):
    """Obtain a DGLGraph for Atoms object
    if neighbor_strategy == "k-nearest":
        edges = nearest_neighbor_edges(
            atoms=atoms,
            cutoff=cutoff,
            max_neighbors=max_neighbors,
            id=id,
            use_canonize=use_canonize,
        )
    else:
        raise ValueError("Not implemented")
    # elif neighbor_strategy == "voronoi"
    #     edges = voronoi_edges(structure)
    ...
```

```
class ALIGNN(nn.Module):
    """Atomistic Line graph network.

    Chain alternating gated graph convolution updates on crystal graph
    and atomistic line graph.
    """

    def __init__(self, config: ALIGNNConfig = ALIGNNConfig(name="alignn")
                 # arrow points here
                 """Initialize class with number of input features, conv layers.')
        super().__init__()
        print(config)
        self.classification = config.classification

        self.atom_embedding = MLPLayer(
            config.atom_input_features, config.hidden_features
        )

        self.edge_embedding = nn.Sequential(
            RBFExpansion(
                vmin=0,
                vmax=8.0, # arrow points here
                bins=config.edge_input_features,
            ),
            MLPLayer(config.edge_input_features, config.embedding_features),
            MLPLayer(config.embedding_features, config.hidden_features),
        )

        self.angle_embedding = nn.Sequential(
            RBFExpansion(
                vmin=-1,
                vmax=1.0,
                bins=config.triplet_input_features,
            ),
            MLPLayer(config.triplet_input_features, config.embedding_features),
            MLPLayer(config.embedding_features, config.hidden_features),
        )
```

```
        self.alignnn_layers = nn.ModuleList(
            [
                ALIGNNConv(
                    config.hidden_features,
                    config.hidden_features,
                )
            for idx in range(config.alignnn_layers)
        ])

        self.gcn_layers = nn.ModuleList(
            [
                EdgeGatedGraphConv(
                    config.hidden_features, config.hidden_features
                )
            for idx in range(config.gcn_layers)
        ])

        self.readout = AvgPooling() # arrow points here

        if self.classification:
            self.fc = nn.Linear(config.hidden_features, 2)
            self.softmax = nn.LogSoftmax(dim=1)
        else:
            self.fc = nn.Linear(config.hidden_features, config.output_features)
        self.link = None
        self.link_name = config.link
        if config.link == "identity":
            self.link = lambda x: x
        elif config.link == "log":
            self.link = torch.exp
            avg_gap = 0.7 # magic number -- average bandgap in dft_3d
            self.fc.bias.data = torch.tensor(
                np.log(avg_gap), dtype=torch.float
            )
        elif config.link == "logit":
            self.link = torch.sigmoid
```

Performance on the Materials Project Dataset

Trained on 69239 materials (DFT data)

#Epochs: 300
Batch_size: 64

Table 1 Performance on the materials project dataset.

Prop	Unit	MAD	CFID	CGCNN	MEGNet	SchNet	ALIGNN	MAD:MAE
E _f	eV/atom	0.93	0.104	0.039	0.028	0.035	0.0221	42.08
E _g	eV	1.35	0.434	0.388	0.33	-	0.218	6.19



- ~44 % improvement by ALIGNN with similar/better training speed
- Similar performance enhancement on QM9 molecule dataset
- Also available on MatBench: <https://matbench.materialsproject.org>

Performance on the JARVIS-DFT Dataset

Property	Units	MAD	CFID	CGCNN	ALIGNN	MAD: MAE
Formation energy	eV(atom) ⁻¹	0.86	0.14	0.063	0.033	26.06
Bandgap (OPT)	eV	0.99	0.30	0.20	0.14	7.07
Total energy	eV(atom) ⁻¹	1.78	0.24	0.078	0.037	48.11
Ehull	eV	1.14	0.22	0.17	0.076	15.00
Bandgap (MBJ)	eV	1.79	0.53	0.41	0.31	5.77
Kv	GPa	52.80	14.12	14.47	10.40	5.08
Gv	GPa	27.16	11.98	11.75	9.48	2.86
Mag. mom	μB	1.27	0.45	0.37	0.26	4.88
SLME (%)	No unit	10.93	6.22	5.66	4.52	2.42
Spillage	No unit	0.52	0.39	0.40	0.35	1.49
Kpoint-length	\AA	17.88	9.68	10.60	9.51	1.88
Plane-wave cutoff	eV	260.4	139.4	151.0	133.8	1.95
ϵ_x (OPT)	No unit	57.40	24.83	27.17	20.40	2.81
ϵ_y (OPT)	No unit	57.54	25.03	26.62	19.99	2.88
ϵ_z (OPT)	No unit	56.03	24.77	25.69	19.57	2.86

Trained on ~55k materials

- Total energy, Formation energy , Ehull
- Bandgap (OPT), Bandgap (MBJ)
- Kv, Gv
- Mag. mom
- ϵ_x (OPT/MBJ), ϵ_y (OPT), ϵ_z (OPT), ϵ (DFPT:elec+ionic)
- Max. piezo. stress coeff (e_{ij})
- Solar-SLME (%)
- Topological-Spillage
- 2D-Exfo. energy
- Kpoint-length
- Plane-wave cutoff
- Max. Electric field gradient
- avg. m_e , avg. m_h
- n-Seebeck, n-PF, p-Seebeck, p-PF

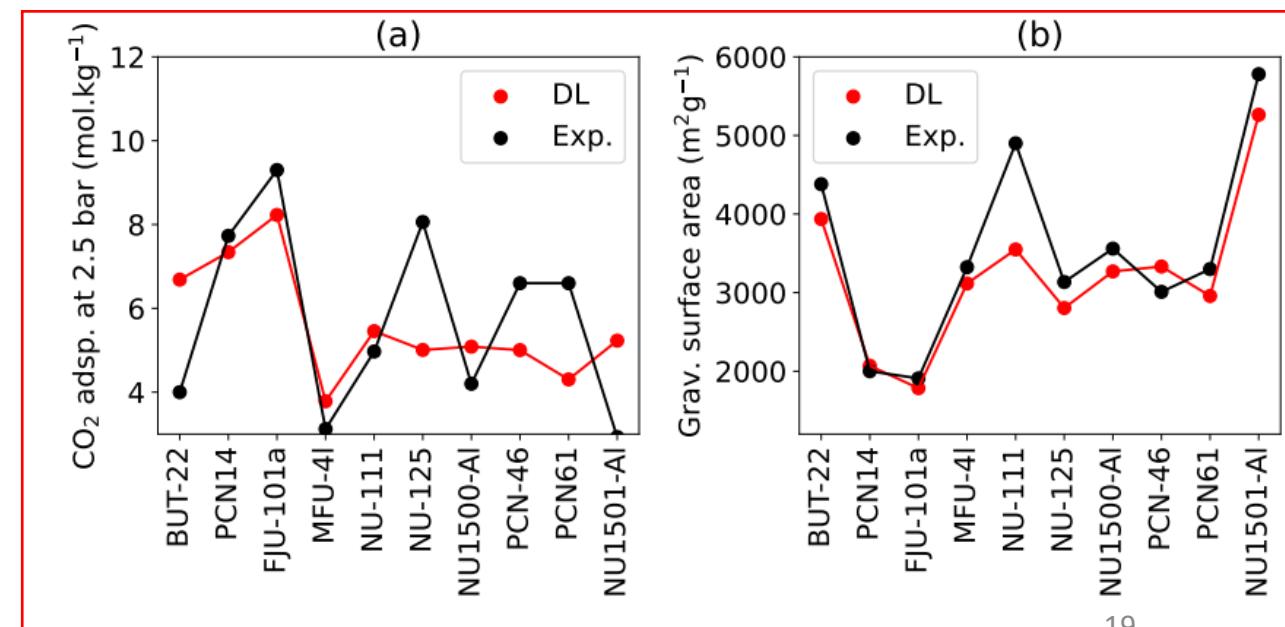
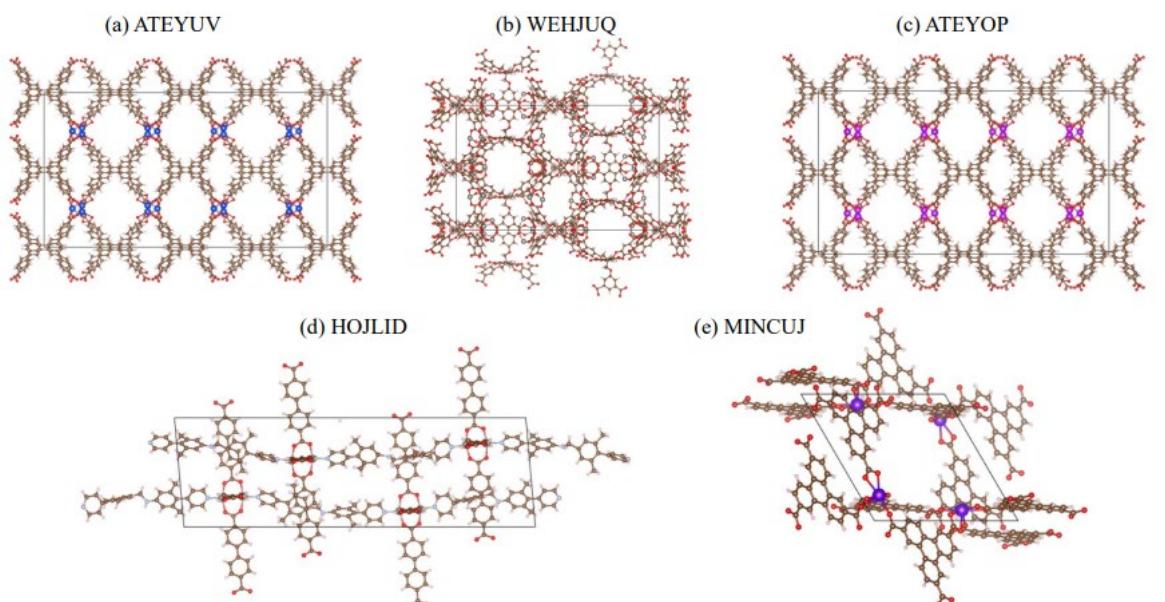
Performance on the QM9 Dataset

Target	Units	SchNet	MEGNet	DimeNet++	ALIGNN
HOMO	eV	0.041	0.043	0.0246	0.0214
LUMO	eV	0.034	0.044	0.0195	0.0195
Gap	eV	0.063	0.066	0.0326	0.0381
ZPVE	eV	0.0017	0.00143	0.00121	0.0031
μ	Debye	0.033	0.05	0.0297	0.0146
α	Bohr ³	0.235	0.081	0.0435	0.0561
R ²	Bohr ²	0.073	0.302	0.331	0.5432
U0	eV	0.014	0.012	0.00632	0.0153
U	eV	0.019	0.013	0.00628	0.0144
H	eV	0.014	0.012	0.00653	0.0147
G	eV	0.014	0.012	0.00756	0.0144

QM9 dataset: 130829 molecules

CO₂ Isotherms: AI for Climate Change

DL model for predicting CO₂ adsorption in MOFs (using hMOF GCMC data)



BCS Superconductors & E-Ph coupling

Superconductors: Materials to conduct electricity without energy loss when they are cooled below a critical temperature, T_c

MgB₂ (T_c = 39 K): Highest T_c ambient condition conventional superconductor

$$T_c = K \Theta_D \exp \left(-\frac{1}{N(0)V} \right)$$

Debye
Temp

DOS at
E_{Fermi}

<https://arxiv.org/abs/2205.00060>

arXiv > cond-mat > arXiv:2205.00060

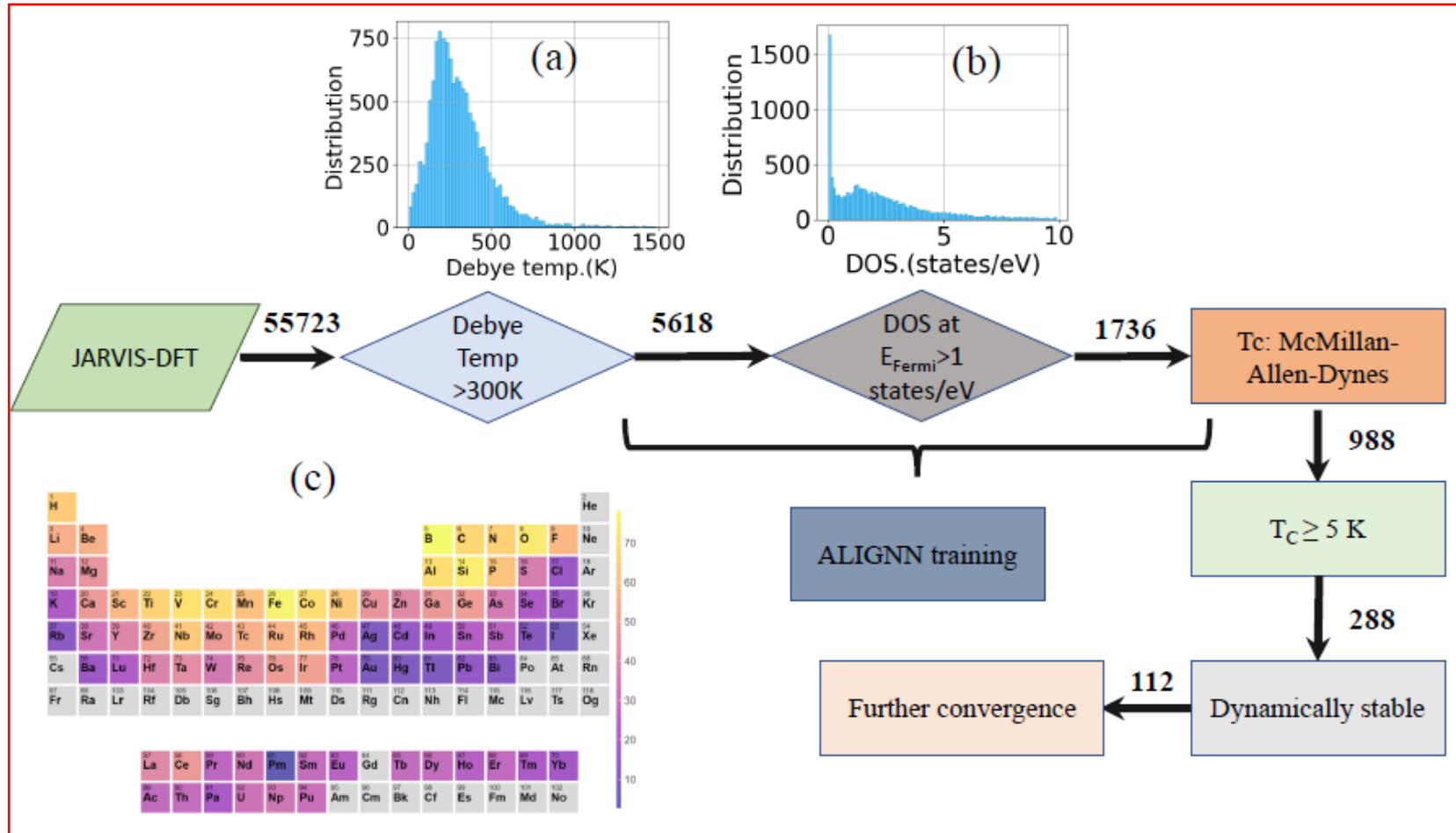
Condensed Matter > Superconductivity

[Submitted on 29 Apr 2022]

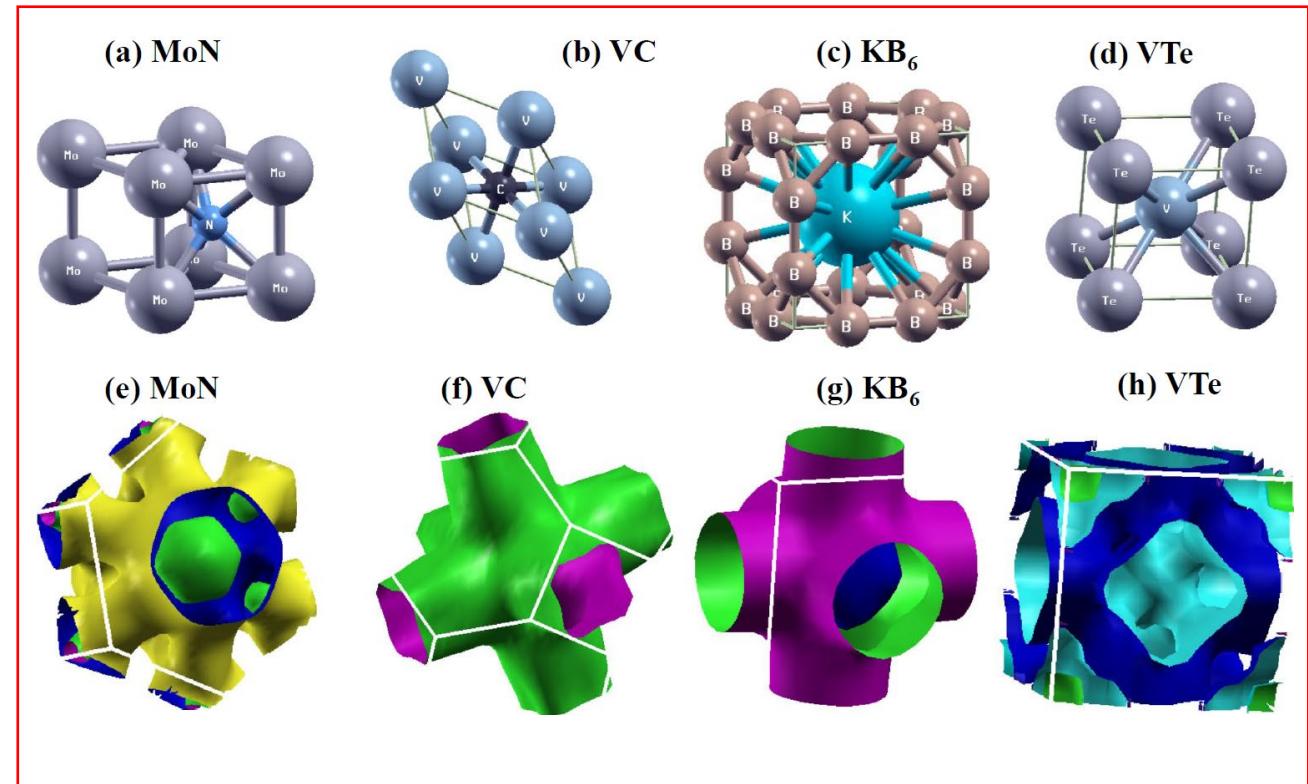
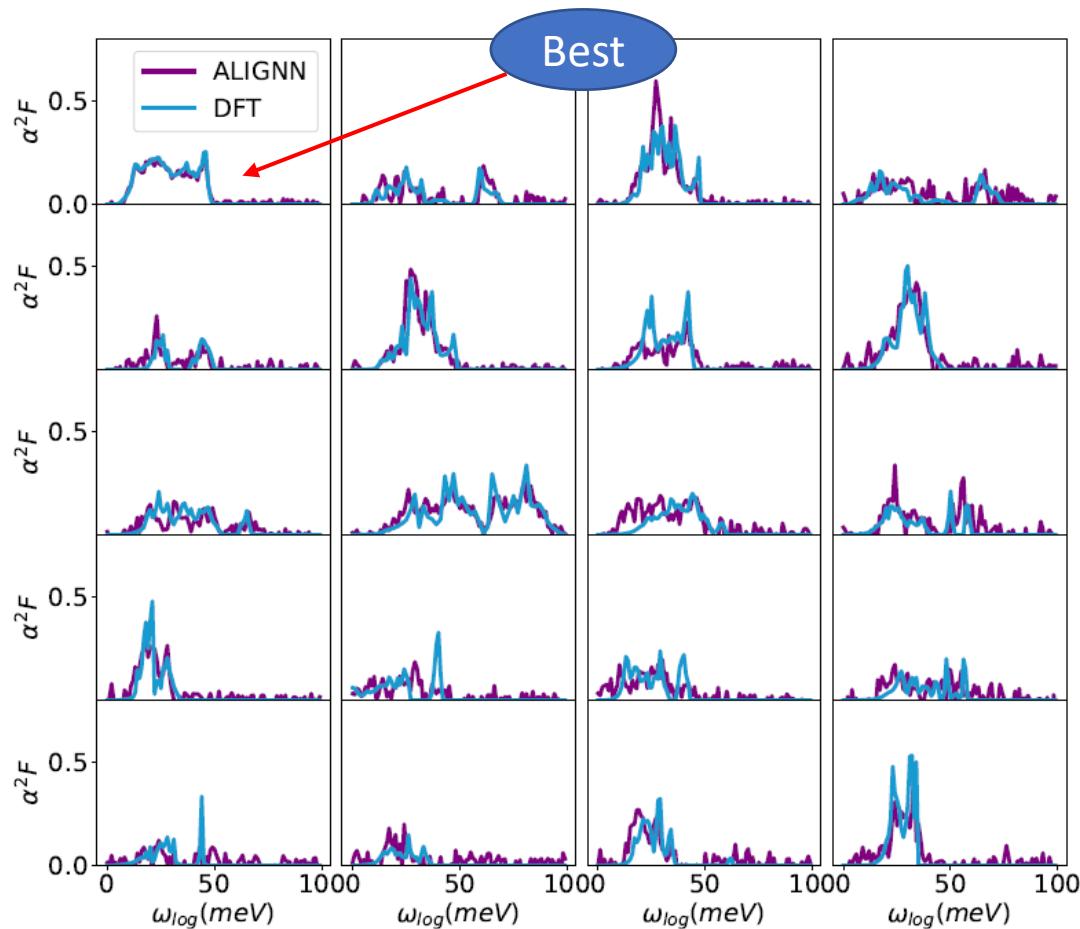
Designing High-Tc Superconductors with BCS-inspired Screening, Density Functional Theory and Deep-learning

Kamal Choudhary, Kevin Garrity

Recent advances in first principles calculations and machine learning techniques allow for a systematic search for phonon-mediated superconductors. We develop a multi-step workflow for the discovery of conventional superconductors, starting with a Bardeen-Cooper-Schrieffer inspired pre-screening of 1736 materials with high Debye temperature and electronic density of states at the Fermi-level. Next, we perform electron-phonon coupling calculations for 988 of them to establish a large and systematic database of BCS superconducting properties. Using the McMillan-Allen-Dynes formula, we identify 112 dynamically stable materials with transition temperatures, T_c > 5 K. In addition, we analyze trends in our dataset and individual materials including MoN, VC, VTe, KB6, Ru3NbC, V3Pt, ScN, LaN₂, RuO₂, and TaC. Finally, we demonstrate that deep-learning models can predict superconductor properties, including the Eliashberg function, thousands of times faster than direct first principles computations. We apply the trained model on the crystallographic open database and pre-screen 8293 candidates for further DFT calculations.



BCS Superconductors



- Prediction on 10 % test data
- 8293 out of 431778 materials in COD as superconductors
- First predicting Eliashberg function, then $T_c \rightarrow 6\%$ improvement
- ALIGNN for both scalar and spectral learning

Unified Force-field



UFF, a full periodic table force field
for molecular mechanics and
molecular dynamics simulations



UFF, a Full Periodic Table Force Field for Molecular Mechanics and Molecular Dynamics Simulations

A. K. Rappé,* C. J. Casewit,[†] K. S. Colwell, W. A. Goddard III,[#] and W. M. Skiff[†]

Contribution from the Department of Chemistry, Colorado State University, Fort Collins,
Colorado 80523. Received March 23, 1992

Abstract: A new molecular mechanics force field, the Universal force field (UFF), is described wherein the force field parameters are estimated using general rules based only on the element, its hybridization, and its connectivity. The force field functional forms, parameters, and generating formulas for the full periodic table are presented.

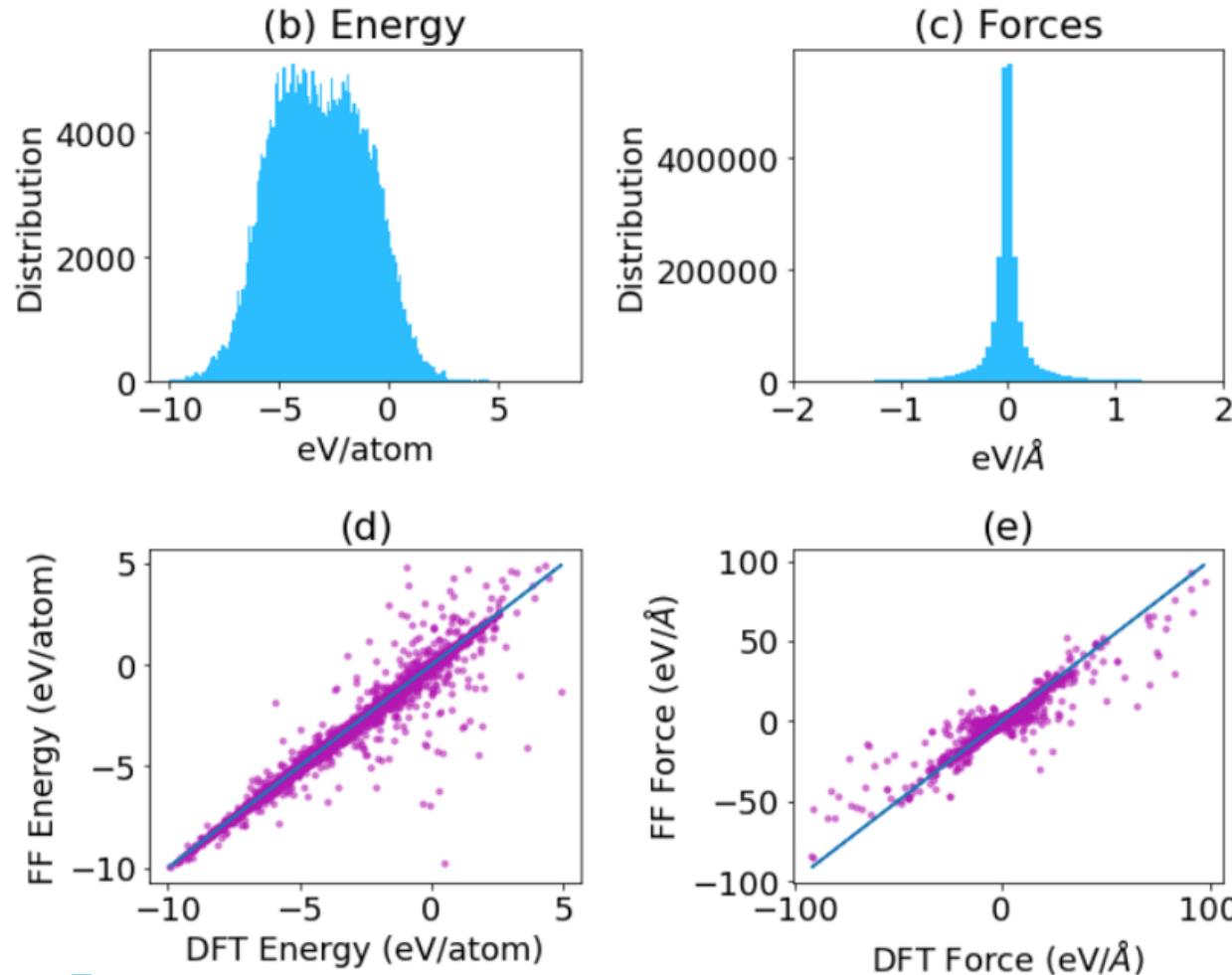
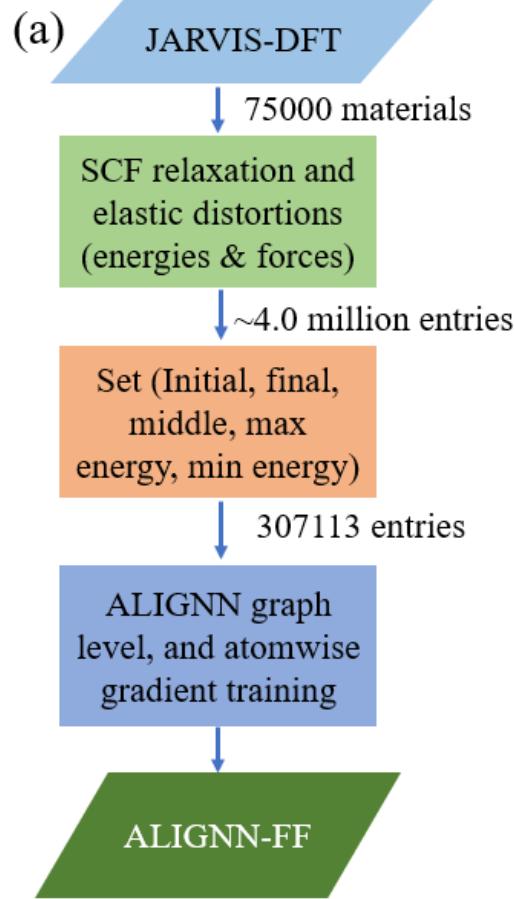
I. Introduction

Parameters and functional forms are the vital infrastructure of molecular mechanics and dynamics force fields. One of the most important uses of molecular dynamics and energy minimization is the estimation of structures for new molecules. Unfortunately, the popular force fields, based on the classic work in the field,^{1a–z} are limited to particular combinations of atoms, for example, those of proteins, organics, or nucleic acids.^{1aa–ff}

metal complex structures are provided in section III. Reference compounds used to obtain covalent radii for the elements are

(1) (a) Bixon, M.; Lifson, S. *Tetrahedron* 1967, 23, 769. (b) Lifson, S. *J. Chim. Phys. Physicochim. Biol.* 1968, 65, 40. (c) Lifson, S.; Warshel, A. *J. Chem. Phys.* 1968, 49, 5116. (d) Levitt, M.; Lifson, S. *J. Mol. Biol.* 1969, 46, 269. (e) Warshel, A.; Levitt, M.; Lifson, S. *J. Mol. Spectrosc.* 1970, 33, 84. (f) Warshel, A.; Lifson, S. *J. Chem. Phys.* 1970, 53, 582. (g) Altona, C.; Sundaralingam, M. *Tetrahedron* 1970, 26, 925. (h) Altona, C.; Sun-

Unified GNN Force-field



Simulate any combination of 89 elements from the periodic table

Digital Discovery



View Article Online
View Journal

PAPER



Unified graph neural network force-field for the periodic table: solid state applications

Cite this: DOI: 10.1039/d2dd00096b

Kamal Choudhary,^{ab} Brian DeCost,^{b,c} Lily Major,^{b,d,e} Keith Butler,^{b,d} Jeyan Thiagaralingam,^{b,d,e} and Francesca Tavazza^{b,c}

Weight	MAE-Energies (eV/atom)	MAE-Forces (eV/Å)
0.1	0.034	0.092
0.5	0.044	0.089
1.0	0.051	0.088
5.0	0.082	0.054
10.0	0.086	0.047

$$l = |E^{DFT} - E^{GNN}| + w \sum_i^{N_{atoms}} |F_i^{DFT} - F_i^{GNN}|$$

Unified GNN Force-field: Forces and stresses

Using r_{ij} , not r_i for equivariance:

$$m_i \frac{d^2 r_i(t)}{dt^2} = \sum_j F_{ij}(t) = -\sum_j \nabla_i U(r_{ij}(t))$$

$$\sigma_{\alpha\beta} = -\frac{1}{V} \sum_i \sum_{i \neq j} F_{ij}^{\alpha} r_{ij}^{\beta} + m_i v_i^{\alpha} v_i^{\beta}$$

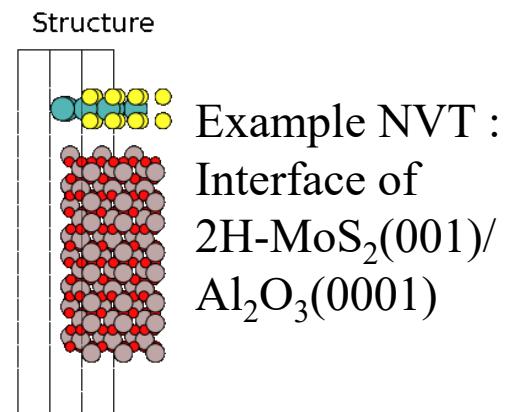
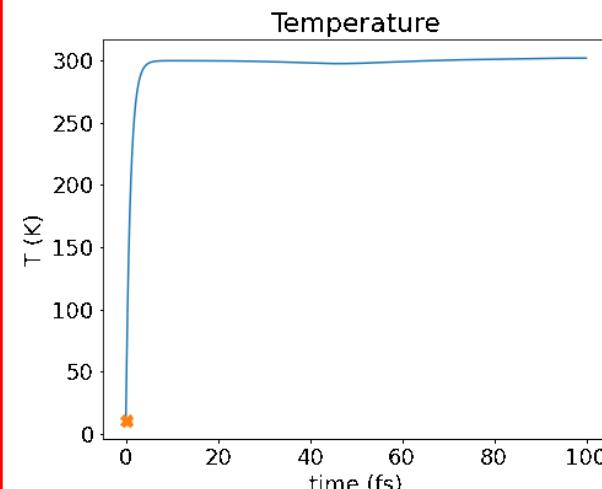
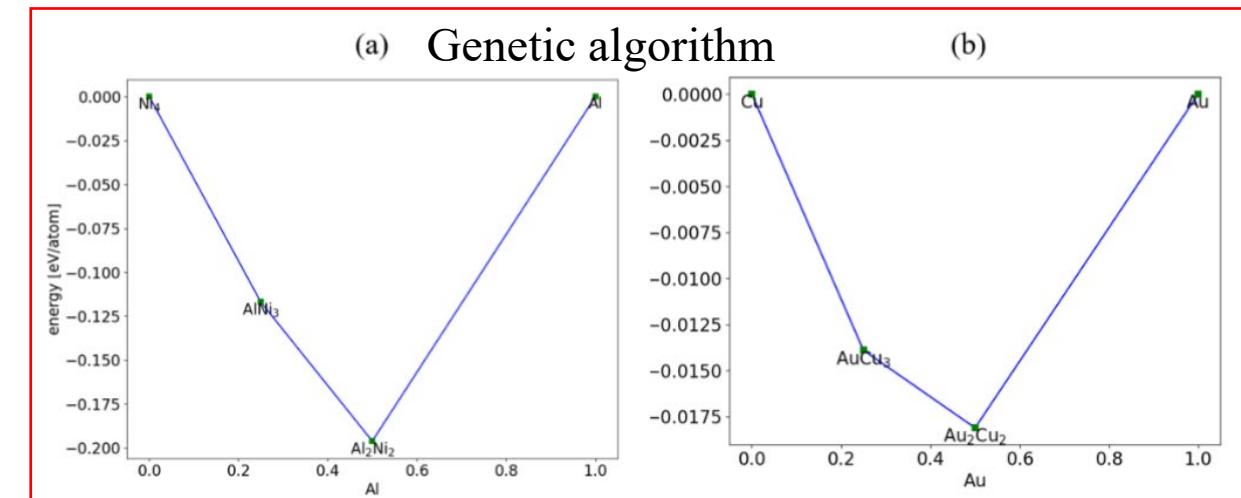
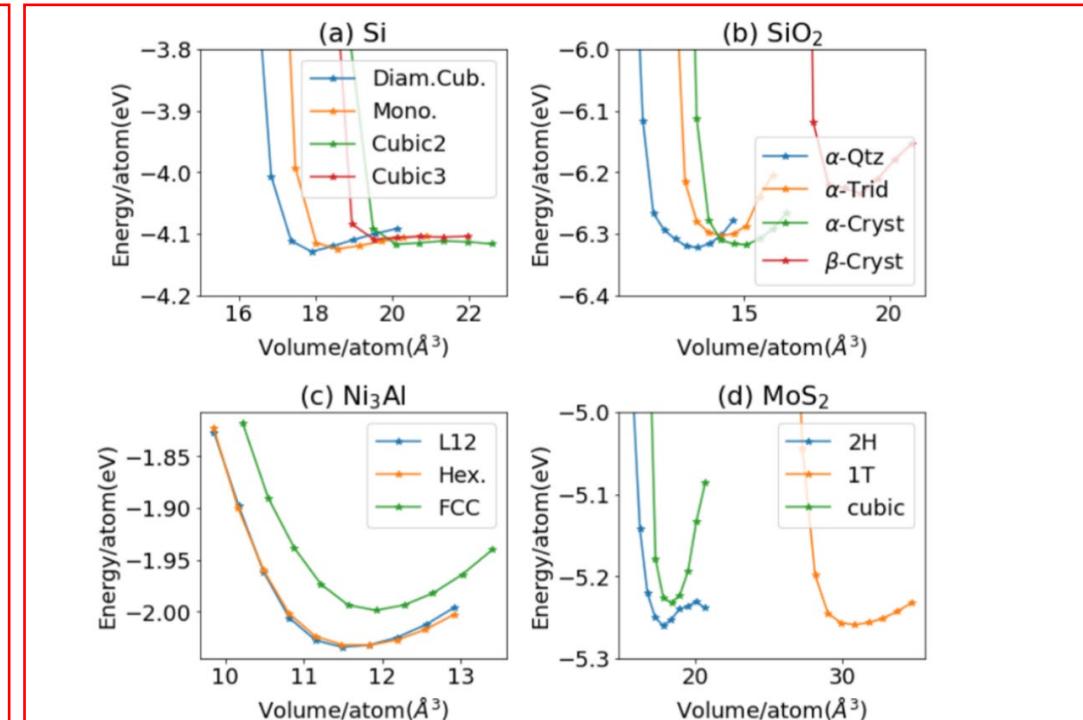
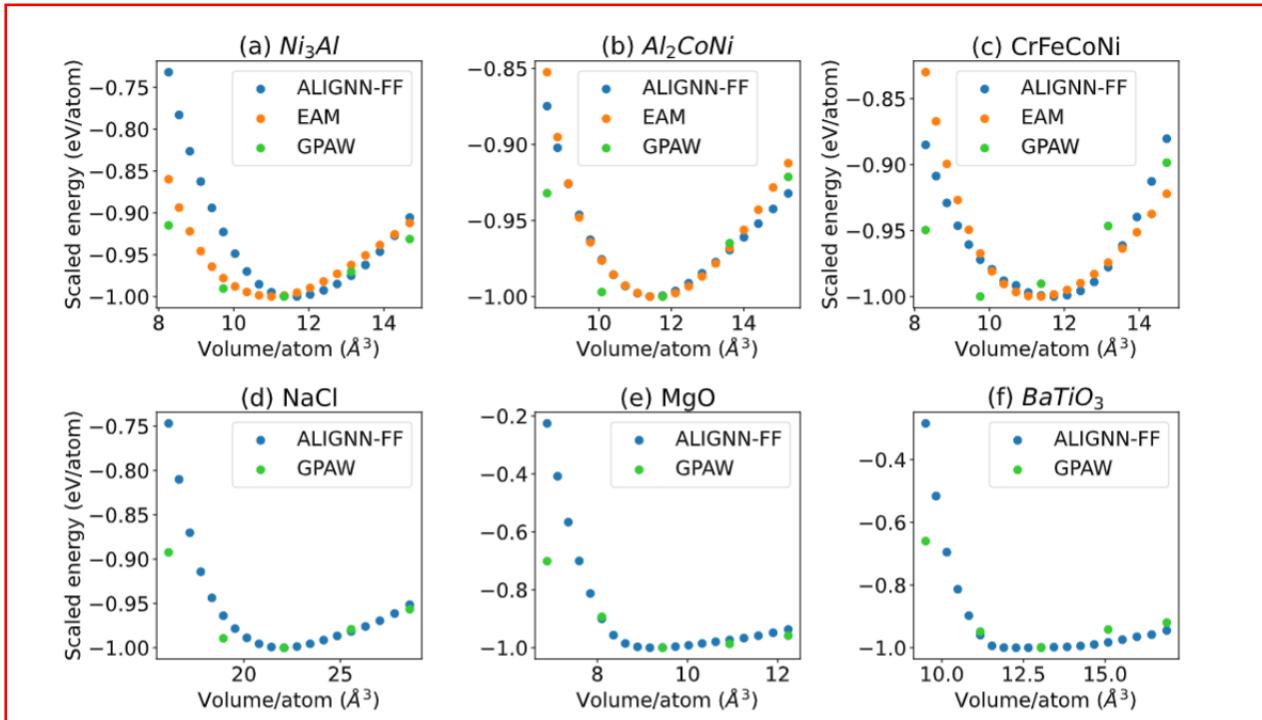
$$l = |E^{DFT} - E^{GNN}| + w \sum_i^{N_{atoms}} |F_i^{DFT} - F_i^{GNN}|$$

Weight	MAE-Energies (eV/atom)	MAE-Forces (eV/Å)
0.1	0.034	0.092
0.5	0.044	0.089
1.0	0.051	0.088
5.0	0.082	0.054
10.0	0.086	0.047

```
if self.config.calculate_gradient:  
    create_graph = True  
  
    dy = (  
        self.config.grad_multiplier  
        * grad( ← PyTorch AutoGrad function  
            # tmp_out,  
            out,  
            r,  
            grad_outputs=torch.ones_like(out),  
            create_graph=create_graph,  
            retain_graph=True,  
        )[0]  
    )  
    g.edata["dy_dr"] = dy  
    g.update_all(fn.copy_e("dy_dr", "m"), fn.sum("m", "gradient"))  
    gradient = torch.squeeze(g.ndata["gradient"])
```

Unified GNN Force-field

EV-curves



Publications using ALIGNNN

npj | computational materials

Explore content ▾ About the journal ▾ Publish with us ▾

nature > npj computational materials > articles > article

Article | Open Access | Published: 15 November 2021

Atomistic Line Graph Neural Network for improved materials property predictions

Kamal Choudhary  & Brian DeCost

npj Computational Materials 7, Article number: 185 (2021) | [Cite this article](#)

10k Accesses | 40 Citations | 19 Altmetric | [Metrics](#)

npj | computational materials

Explore content ▾ About the journal ▾ Publish with us ▾

nature > npj computational materials > articles > article

Article | Open Access | Published: 22 November 2022

Designing high- T_C superconductors with BCS-inspired screening, density functional theory, and deep-learning

Kamal Choudhary  & Kevin Garrity

npj Computational Materials 8, Article number: 244 (2022) | [Cite this article](#)

1572 Accesses | 2 Citations | 7 Altmetric | [Metrics](#)

Prediction of the Electron Density of States for Crystalline Compounds with Atomistic Line Graph Neural Networks (ALIGNNN)

Prathik R. Kaundinya, Kamal Choudhary & Surya R. Kalidindi 

Digital Discovery

PAPER

 Check for updates

Cite this: DOI: 10.1039/d2dd00096b



[View Article Online](#)
[View Journal](#)

Unified graph neural network force-field for the periodic table: solid state applications

Kamal Choudhary,  ^{a,b} Brian DeCost,  ^c Lily Major,  ^{de} Keith Butler,  ^e Jeyan Thiagaralingam  ^e and Francesca Tavazza 



Computational Materials Science

Volume 210, July 2022, 111388



Full length article

Graph neural network predictions of metal organic framework CO₂ adsorption properties

Kamal Choudhary  ^{a,b,c} , Taner Yildirim ^d, Daniel W. Siderius ^e, A. Gilad Kusne ^f, Austin McDannald ^f, Diana L. Ortiz-Montalvo ^f

PHYSICAL REVIEW MATERIALS

Highlights Recent Accepted Research Updates Collections Authors Referees Search

Rapid prediction of phonon structure and properties using the atomistic line graph neural network (ALIGNNN)

Ramya Gurunathan, Kamal Choudhary, and Francesca Tavazza
Phys. Rev. Materials 7, 023803 – Published 24 February 2023

arXiv > cond-mat > arXiv:2205.08366

Condensed Matter > Materials Science

[Submitted on 17 May 2022]

A Deep-learning Model for Fast Prediction of Vacancy Formation in Diverse Materials

Kamal Choudhary, Bobby G. Sumpter

AtomVision

A deep learning framework for atomistic image data

usnistgov / atomvision Public

generated from usnistgov/opensource-repo

<> Code Issues Pull requests Projects Wiki Security Insights Settings

master ▾ 1 branch 0 tags

knc6 Add github action.

.github Add github action

atomvision Remove unnecessary

CODEMETA.yaml Initial commit

LICENSE.md Initial commit

README.md Update README.n

setup.py Fix requirements.

README.md

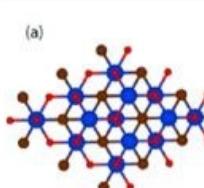
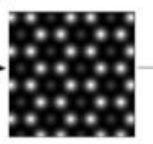
Open in Colab

JCIM JOURNAL OF CHEMICAL INFORMATION AND MODELING
pubs.acs.org/jcim Article

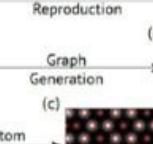
AtomVision: A Machine Vision Library for Atomistic Images
Kamal Choudhary,* Ramya Gurunathan, Brian DeCost, and Adam Biacchi

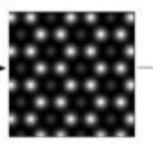
Cite This: <https://doi.org/10.1021/acs.jcim.2c01533> Read Online

ACCESS | Metrics & More | Article Recommendations

(a)  Contrast Model → (b) 

Autoencoder Reproduction → (e) 

Graph Generation → (d) 

Atom Localization → (c) 

27

Notebook

A screenshot of a GitHub repository page for the file `AtomVisionExample.ipynb`. The page includes a lock icon, the URL `github.com/knc6/jarvis-tools-notebooks/blob/master/jarvis-tools-notebooks/AtomVisionExample.ipynb`, and a 'Copilot' button. Below the URL, it says 'Created using Colaboratory'. The repository name is 'JARVIS-Materials-Design / jarvis-tools-notebooks'. The main content area shows the notebook's code, blame information, and line count (1083 lines, 1083 loc). A 'Code' tab is selected. At the bottom, there are links for 'Issues', 'Pull requests', 'Actions', and 'Projects'. A 'Open in Colab' button is also present.

Example to run AtomVision

AtomVision is a deep learning package to perform various operations on image data including segmentation and classification of ima

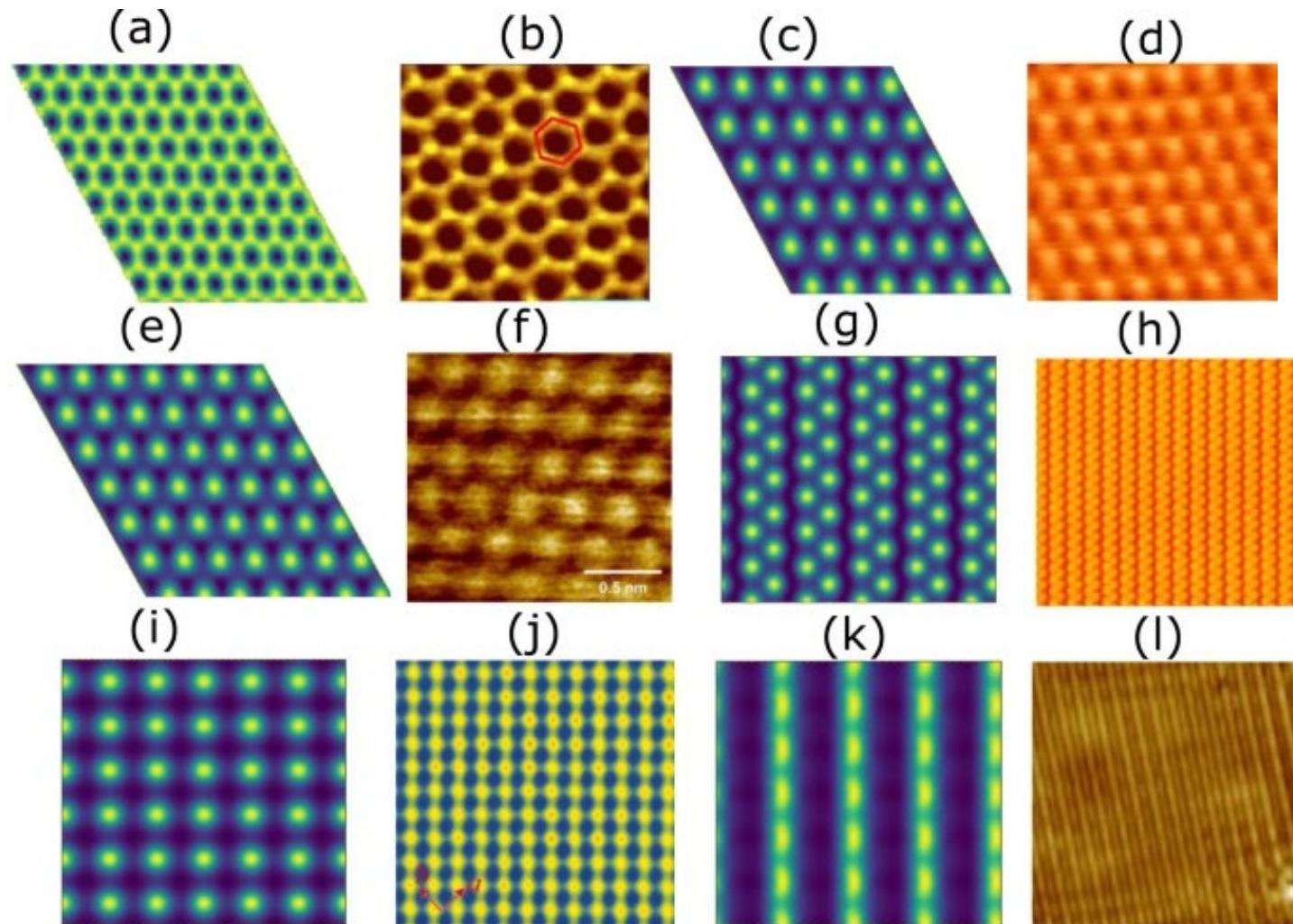
Table of Contents

1. Install AtomVision package
2. Generating STEM image with convolution approximation: graphene example
3. Train an autoencoder
4. Train a Generative Adversarial Network
5. Train DenseNet classification model on JARVIS-DFT 2D STEM image dataset
6. Train ALIGNN based classification model on JARVIS-DFT 2D STEM image dataset

Get the repository and install

In [1]: `pip install atomvision`

Scanning Tunneling Microscope Image



$$I_t \propto \rho(\mathbf{r}, E_F) \equiv \sum_{\mu} |\psi_{\mu}(\mathbf{r})|^2 \delta(\varepsilon_{\mu} - E_F)$$

Tersoff-Hamann Approach

[nature](#) > [scientific data](#) > [data descriptors](#) > [article](#)

Data Descriptor | [Open Access](#) | Published: 11 February 2021

Computational scanning tunneling microscope image database

Kamal Choudhary , Kevin F. Garrity, Charles Camp, Sergei V. Kalinin, Rama Vasudevan, Maxim Ziatdinov & Francesca Tavazza

[Scientific Data](#) **8**, Article number: 57 (2021) | [Cite this article](#)

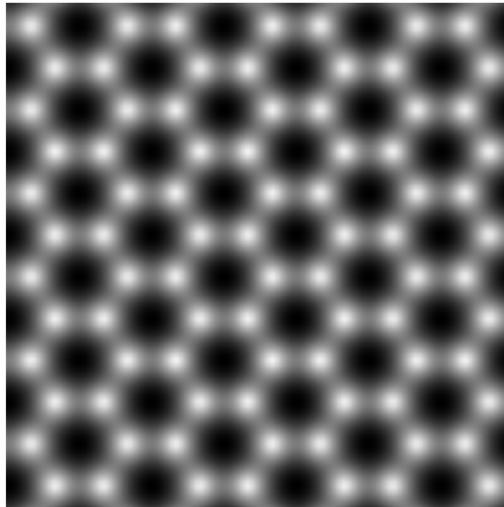
1594 Accesses | **1** Citations | **4** Altmetric | [Metrics](#)



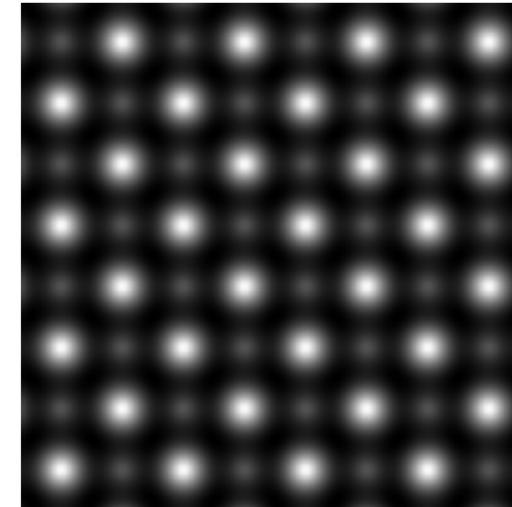
Scanning Transmission Electron Microscope Image

Convolution approximation: accurate for **thin films** mainly (here 2D mats.)
Based on Rutherford scattering model

$$I(\mathbf{r}) = R(\mathbf{r}, Z) \otimes \text{PSF}(\mathbf{r}), \quad R(\mathbf{r}, Z) = \sum_{i=1}^N Z_i^{1.7} \delta(\mathbf{r} - \mathbf{r}_i)$$



C: JVASP-667



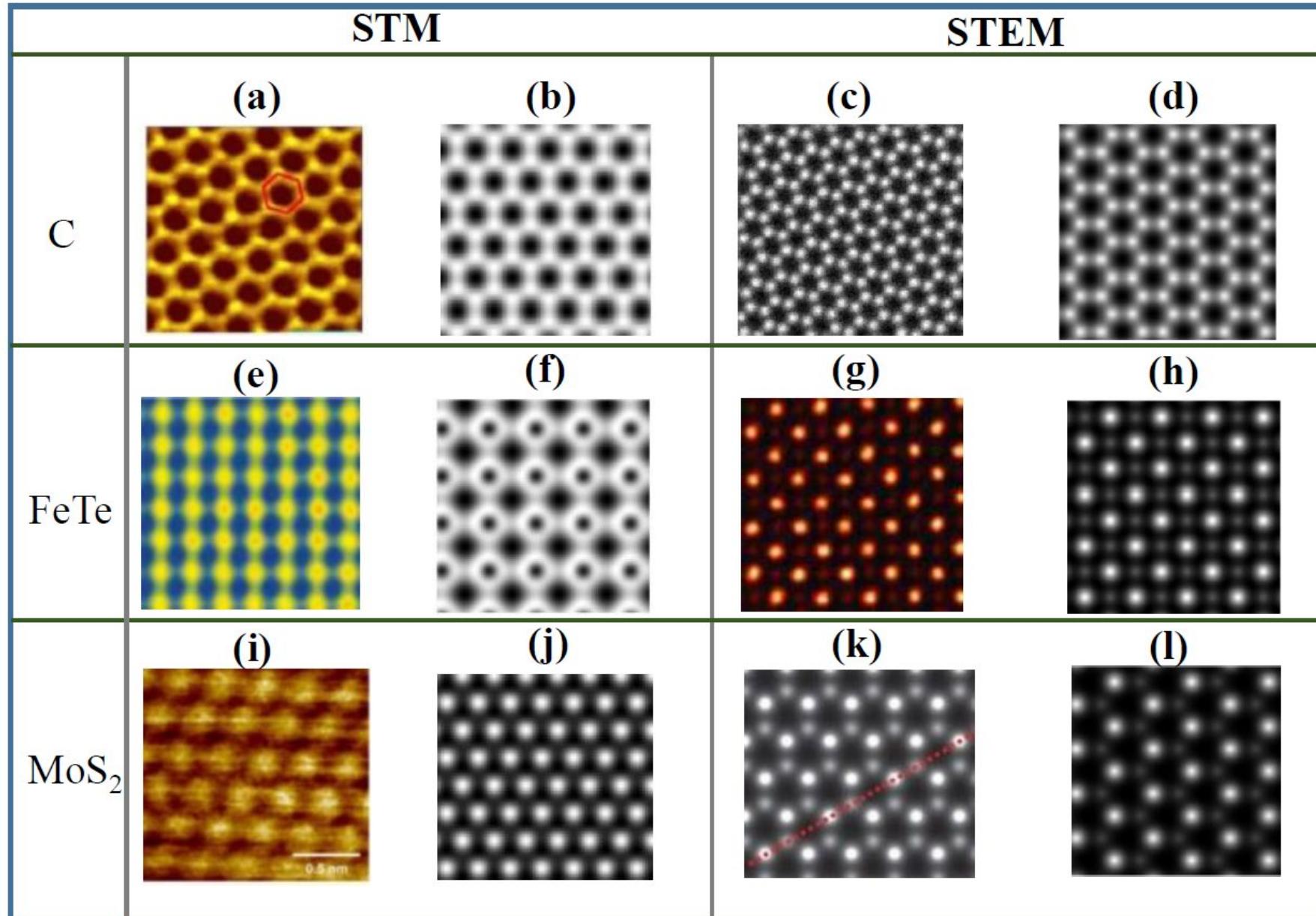
FeTe: JVASP-6667



PPdSe: JVASP-6316

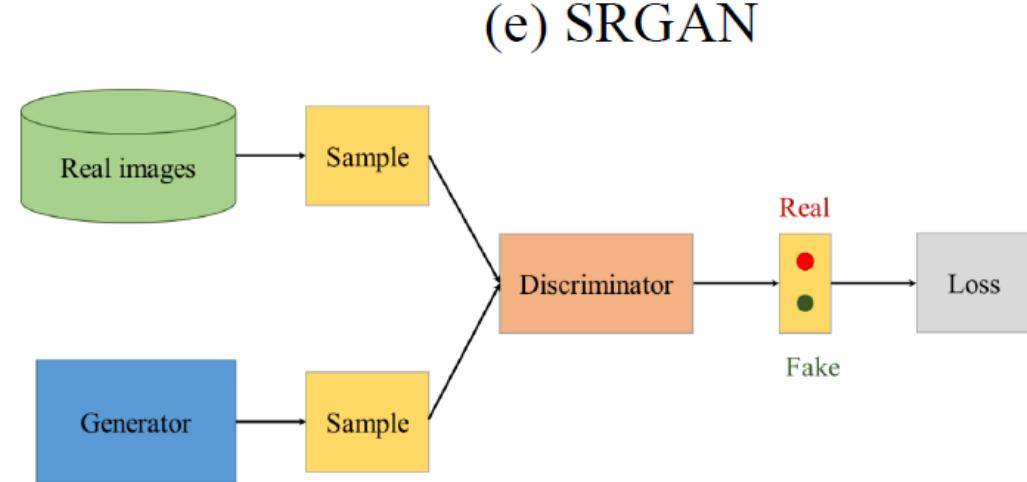
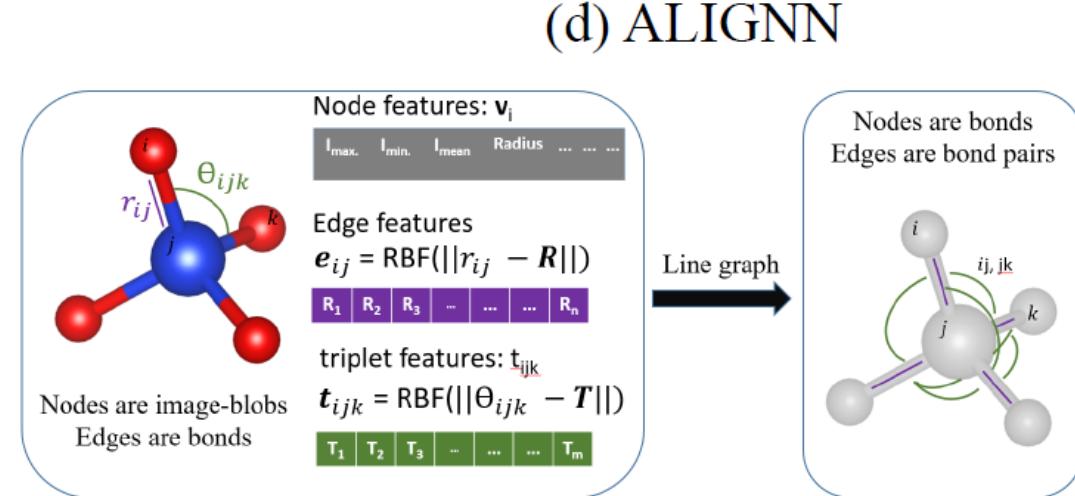
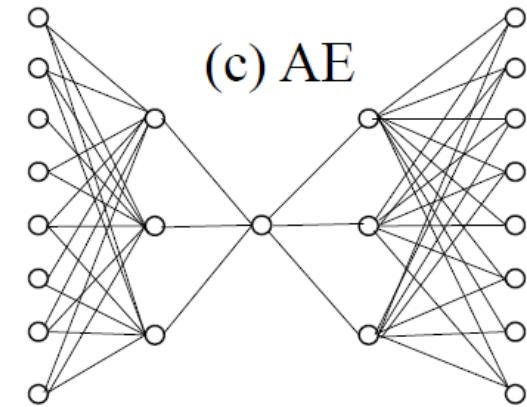
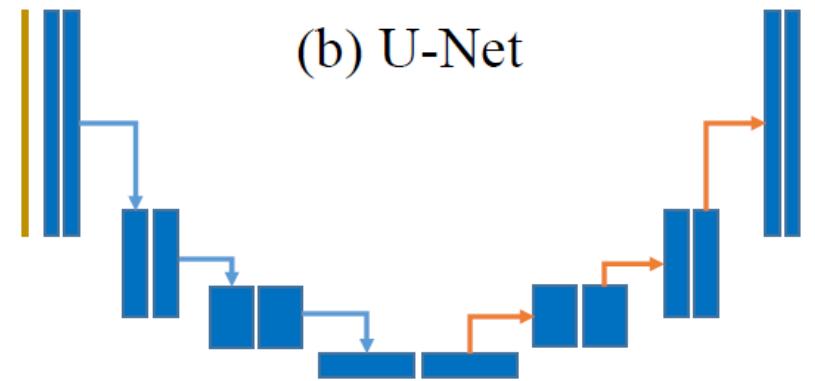
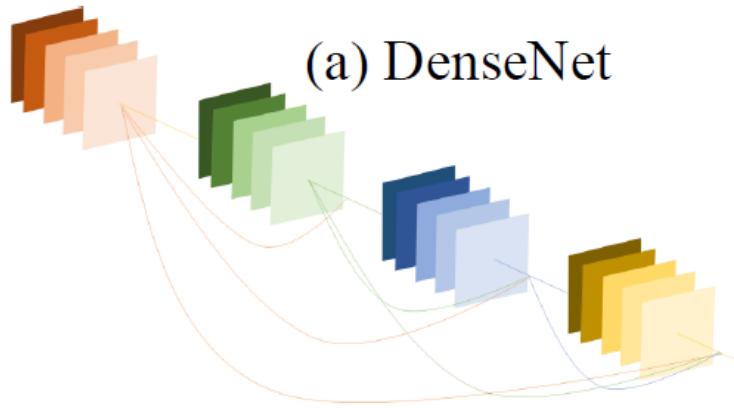


Experimental vs computational images





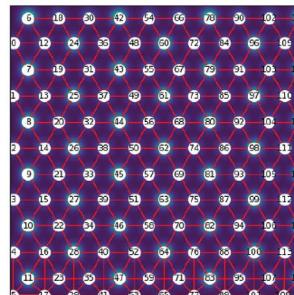
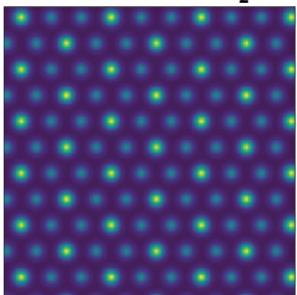
Deep Learning architectures in AtomVision



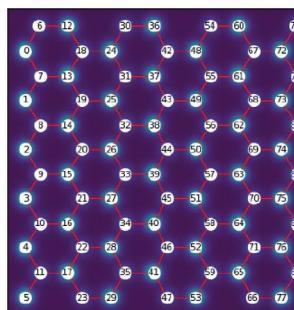
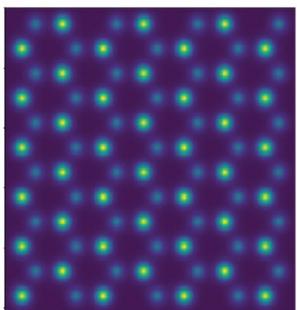


Graph construction

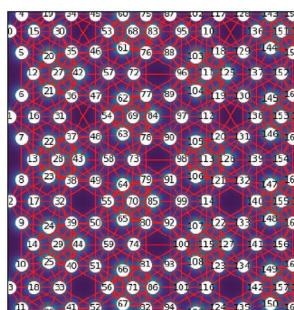
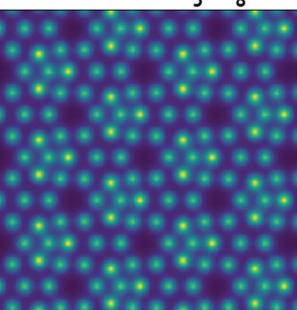
PbI₂: JVASP-76548



GaTe: JVASP-6838



Nb₃Br₈: JVASP-76350



master atomvision / atomvision / scripts / train_classifier_alignn.py

Code Blame 388 lines (335 loc) · 10.8 KB

```
95     def labelled_images_to_graphs(images, labels, border_pxl=0, saveto=""):  
96         """  
97             Get labelled image to graph.  
98  
99         Args:  
100             images: list of image arrays.  
101  
102             labels: list of corresponding class labels.  
103             """  
104         graphs = []  
105         line_graphs = []  
106         n = 0  
107         for img, lbl in zip(images, labels):  
108             if border_pxl != 0:  
109                 img = crop_image(img, border_pxl)  
110                 blob_list = get_blob_positions(img)  
111                 g, lg = blob_list_to_graph(img, blob_list)  
112                 graphs.append(g)  
113                 line_graphs.append(lg)  
114             n = n + 1  
115         return graphs, line_graphs
```

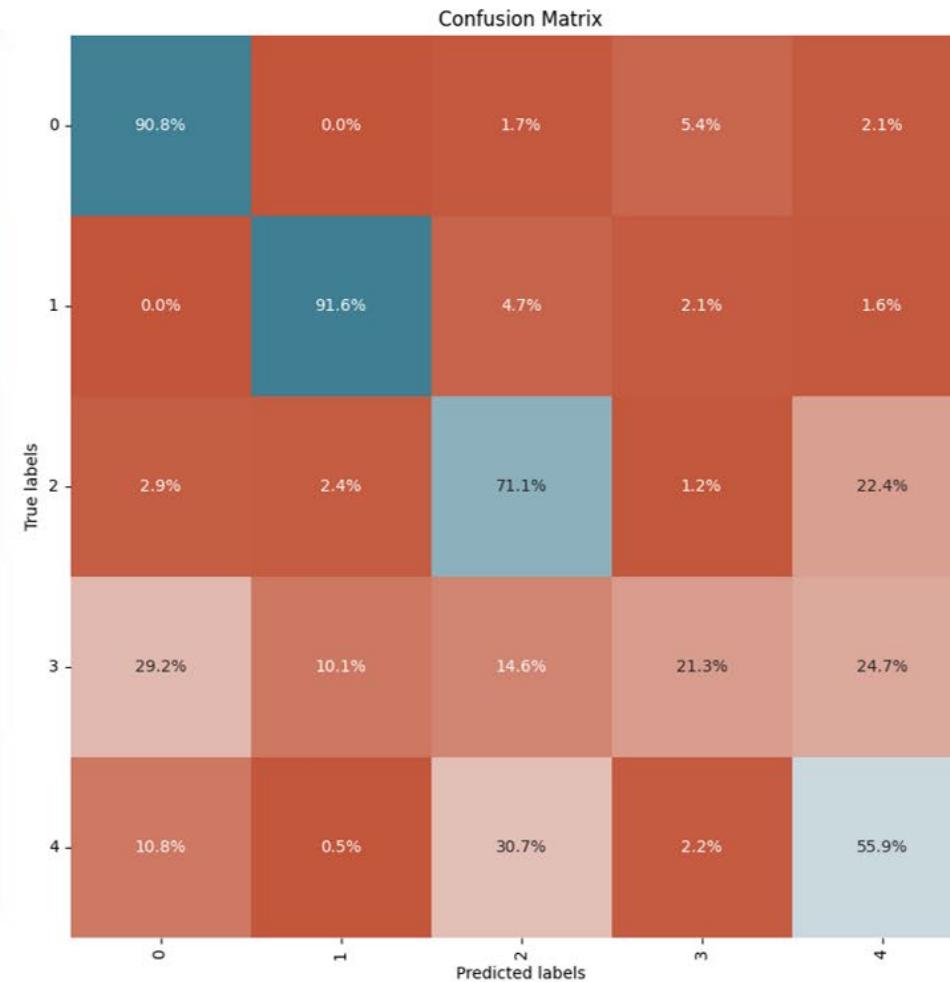
A red arrow points from the word "blob_list" in line 110 of the code to the corresponding line in the "get_blob_positions" function definition.

Image classification (CNN & GNN)

(a) DenseNet Classifier

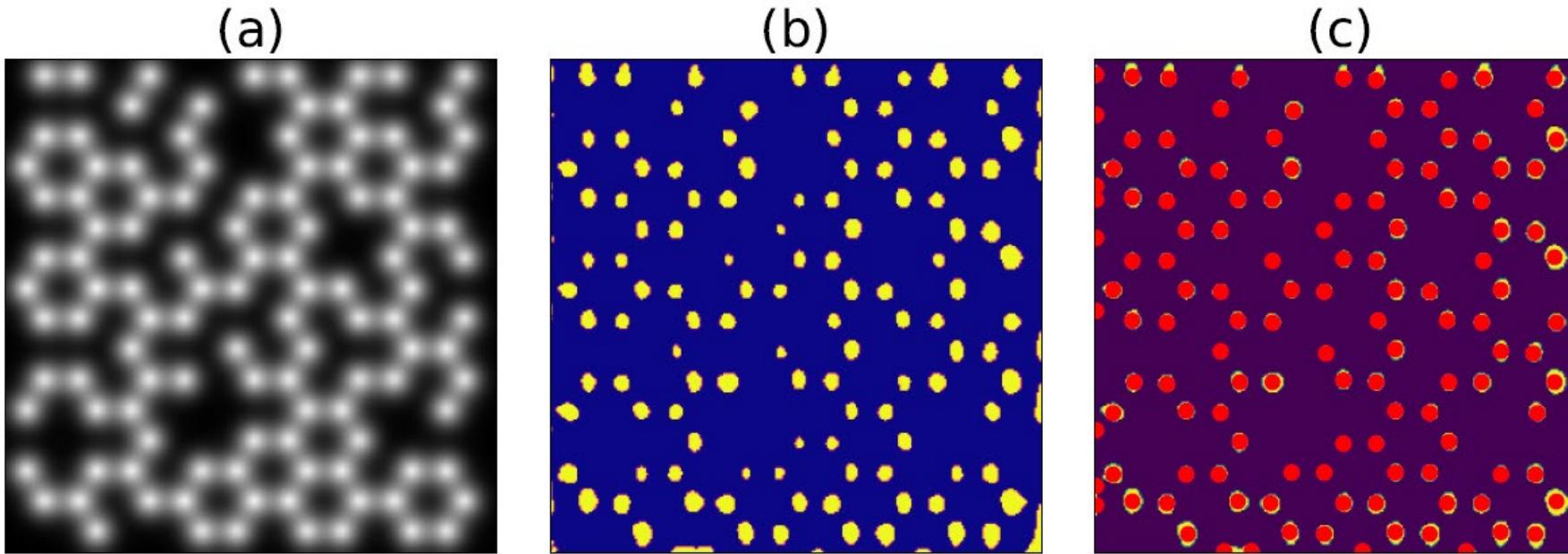


(b) ALIGNN Classifier



2D Bravais lattice classification (DenseNet (82%), ALIGNN(78%)):
0) hexagonal, 1) square, 2) rectangle, 3) rhombus, 4) parallelogram
Baseline accuracy $1/5 = 20\%$

Semantic segmentation



Semantic segmentation using U-Net:

- Atom vs background, pixelwise classification
- Train on computational dataset, fine-tune for experiments

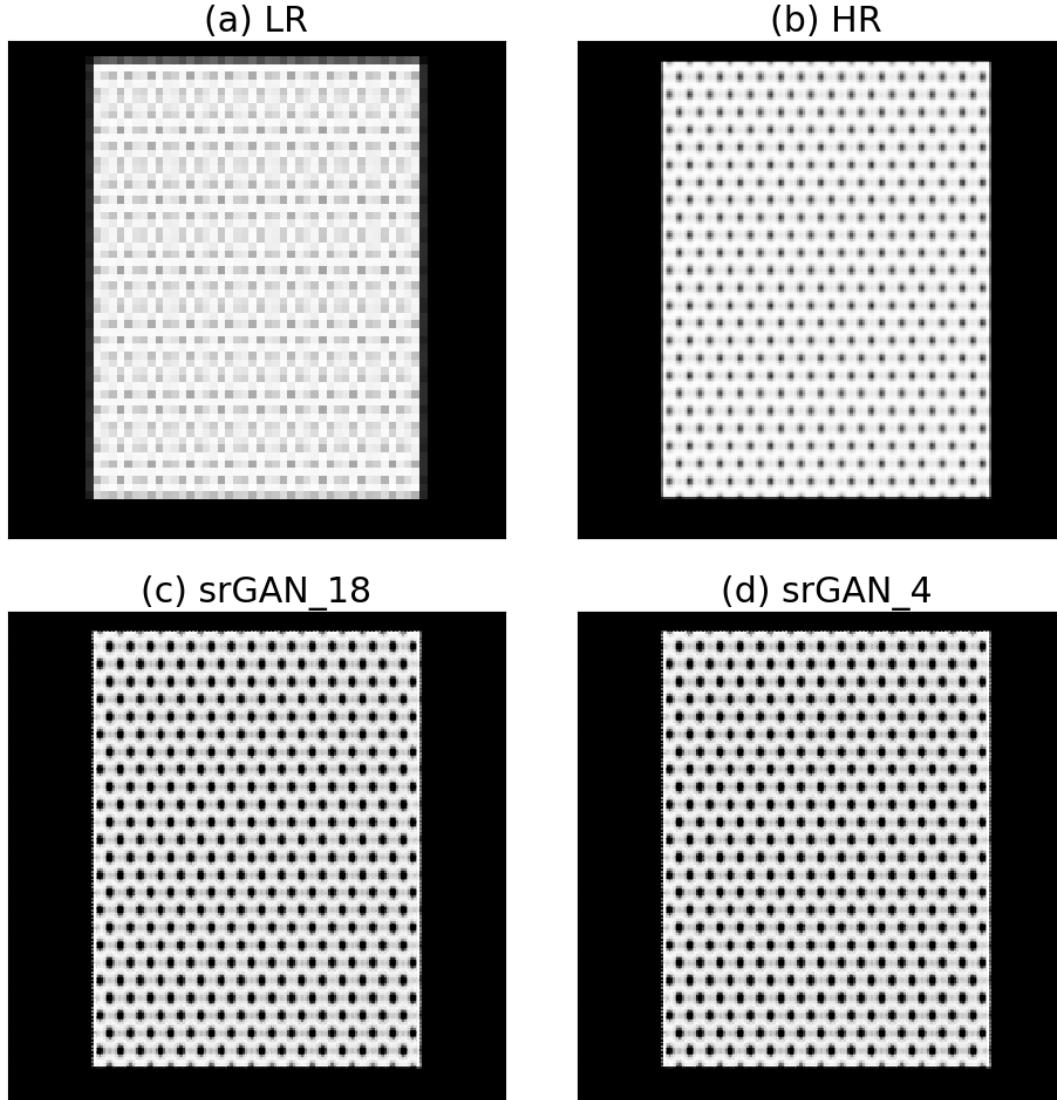
Classification + Localization



<http://cs231n.stanford.edu/syllabus.html>

Super-resolution

SR-GAN for enhancing image resolution with VGG19



ChemNLP

A Natural Language Processing based Library for Materials Chemistry Text Data

The screenshot shows the GitHub repository page for `usnistgov/chemnlp`. The repository is public. The navigation bar includes links for Code, Issues, Pull requests, Actions, Projects, Wiki, and Security. Below the navigation bar, there are buttons for switching branches (main), viewing 3 branches, and viewing 0 tags. There are also buttons for Go to file and Add file. A recent commit by `knc6` is shown, which merged pull request #6 from `usnistgov/develop` and was committed 6 days ago with the hash `68cef7b`. At the bottom, the arXiv logo is present along with the category `Condensed Matter > Materials Science`, the submission date `[Submitted on 17 Sep 2022]`, and search and help links.

Code Issues Pull requests Actions Projects Wiki Security

main ▾ 3 branches 0 tags Go to file Add file

knc6 Merge pull request #6 from usnistgov/develop ... ✓ 68cef7b 6 days ago

arXiv > cond-mat > arXiv:2209.08203

Condensed Matter > Materials Science

[Submitted on 17 Sep 2022]

ChemNLP: A Natural Language Processing based Library for Materials Chemistry Text Data

Notebook

The screenshot shows a GitHub repository page for 'JARVIS-Materials-Design/jarvis-tools-notebooks'. The repository is public and has 16 forks. The 'Code' tab is selected. A specific notebook, 'ChemNLP_Example.ipynb', is displayed. The notebook was created by 'knc6' using Colaboratory. It contains 780 lines of code (780 loc) and is 36.9 KB in size. There is a 'Preview' button, a 'Code' button, and a 'Blame' button. Below the preview area is a 'CO' icon followed by the text 'Open in Colab'. The main content area features a section titled 'Example to run ChemNLP' with the following text: 'ChemNLP is a Natural Language Processing based Library for Materials Chemistry Text Data.' Below this is a code cell starting with 'In []:' and containing the command 'pip install chemnlp'.

← → ⌂ github.com/JARVIS-Materials-Design/jarvis-tools-notebooks/blob/master/jarvis-tools-notebooks/ChemNLP_Example.ipynb ↵

Search or jump to... / Pull requests Issues Codespaces Marketplace Explore

JARVIS-Materials-Design / jarvis-tools-notebooks Public Edit Pins Watch 3 Fork 16 Insights Settings

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

jarvis-tools-notebooks / jarvis-tools-notebooks / ChemNLP_Example.ipynb

knc6 Created using Colaboratory 9f5f43a · 5 r

Preview Code Blame 780 lines (780 loc) · 36.9 KB R

Open in Colab

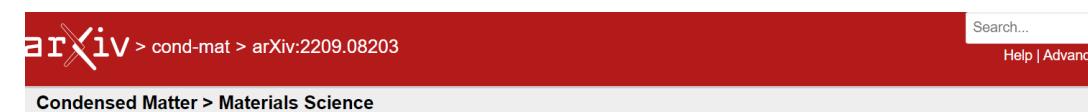
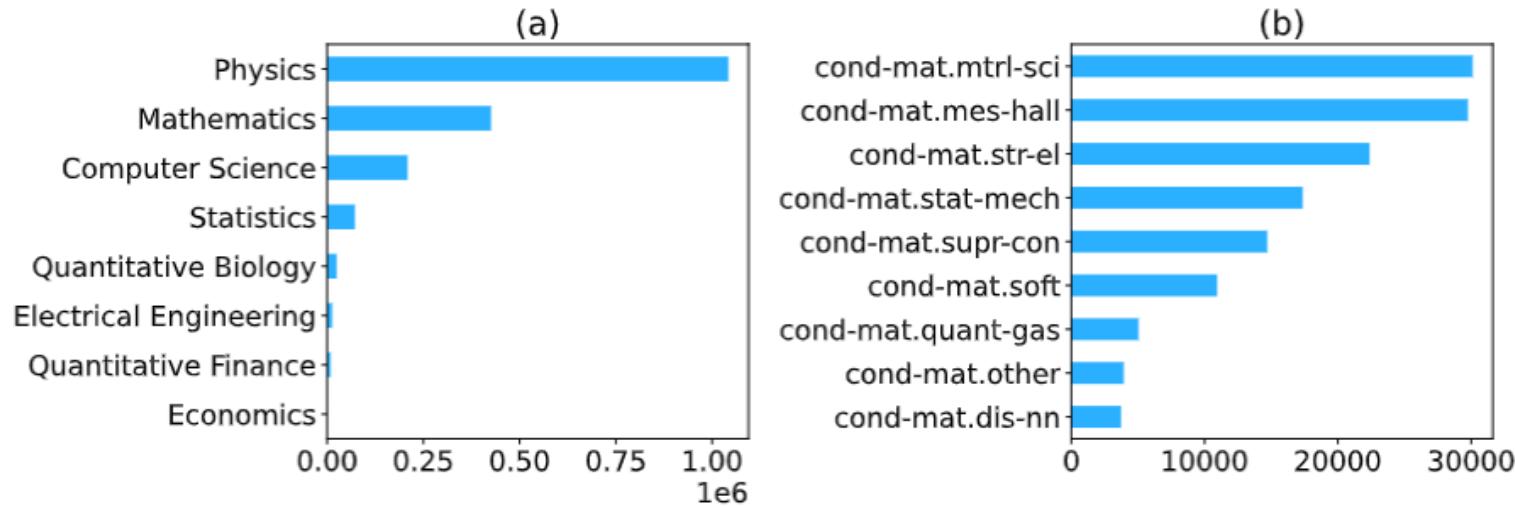
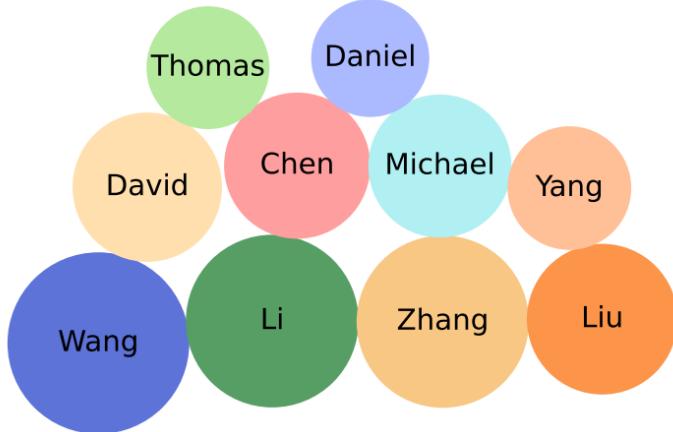
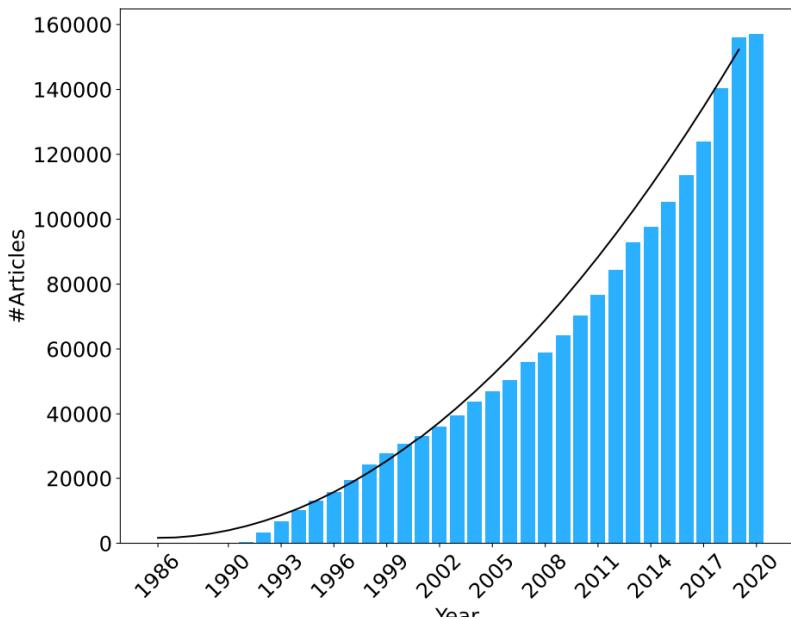
Example to run ChemNLP

ChemNLP is a Natural Language Processing based Library for Materials Chemistry Text Data.

In []: pip install chemnlp

ChemNLP

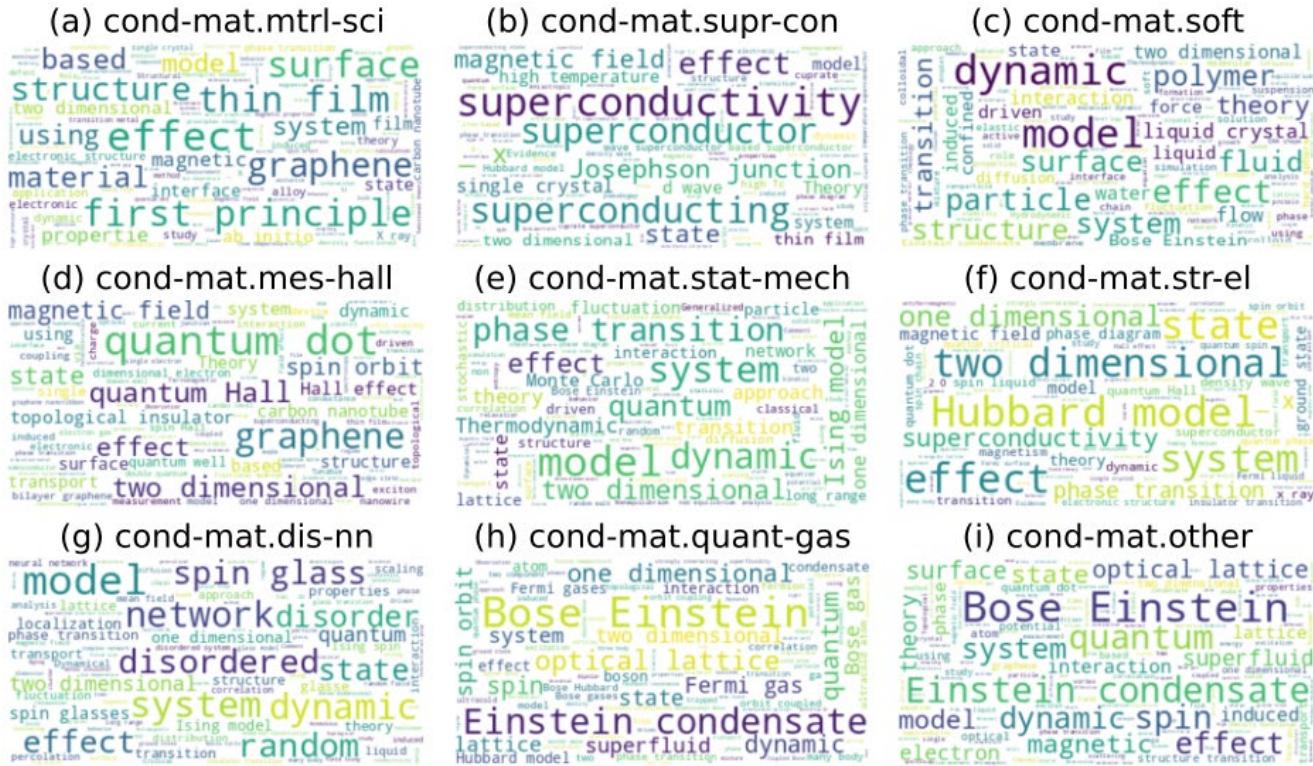
arXiv dataset 1.8 million articles



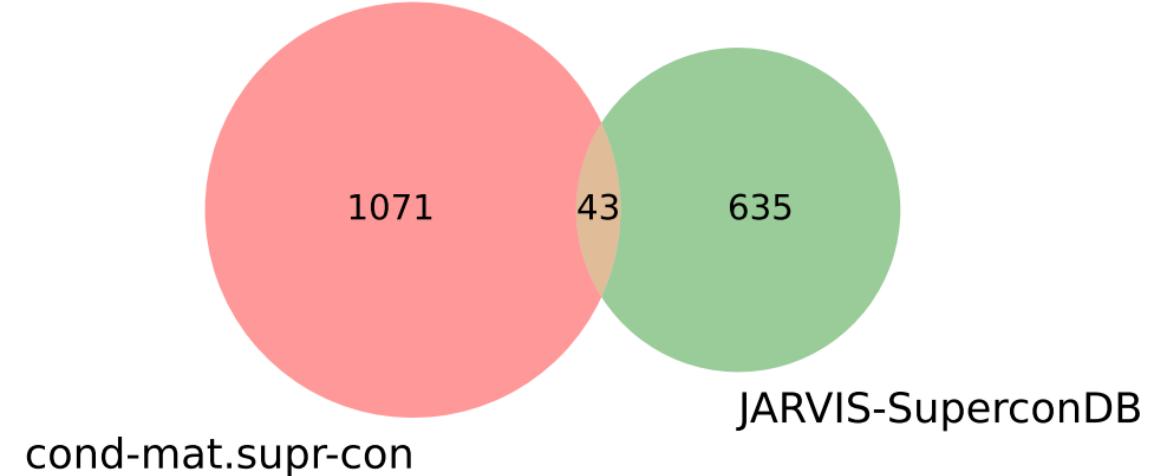
Natural language processing (NLP) has an immense potential to aid materials design processes. While there have been several advancements in this field, a complete and integrated framework with well-curated dataset and tools to apply NLP is still needed. In this work, we present the ChemNLP library and an accompanying web-app that can be used to analyze important materials chemistry information. We use the publicly available arXiv dataset that has been collected over 34 years and contains ~1.8 million articles. First, we analyze the article publication trend, categorizations, and common phrases in the arXiv dataset. Then, we develop a user-friendly, interactive web-app to retrieve articles for a given chemical compound. Furthermore, we demonstrate the effectiveness of the proposed framework to accelerate the identification of superconducting materials. We determine the overlap between density functional theory and text-based databases for superconductors. Finally, we perform machine learning based clustering and classification tasks to quickly categorize scholarly articles given article title information with accuracy up to 81.2 %. ChemNLP is available at the websites: this [https URL](#) and this [https URL](#).

arXiv:2209.08203

ChemNLP

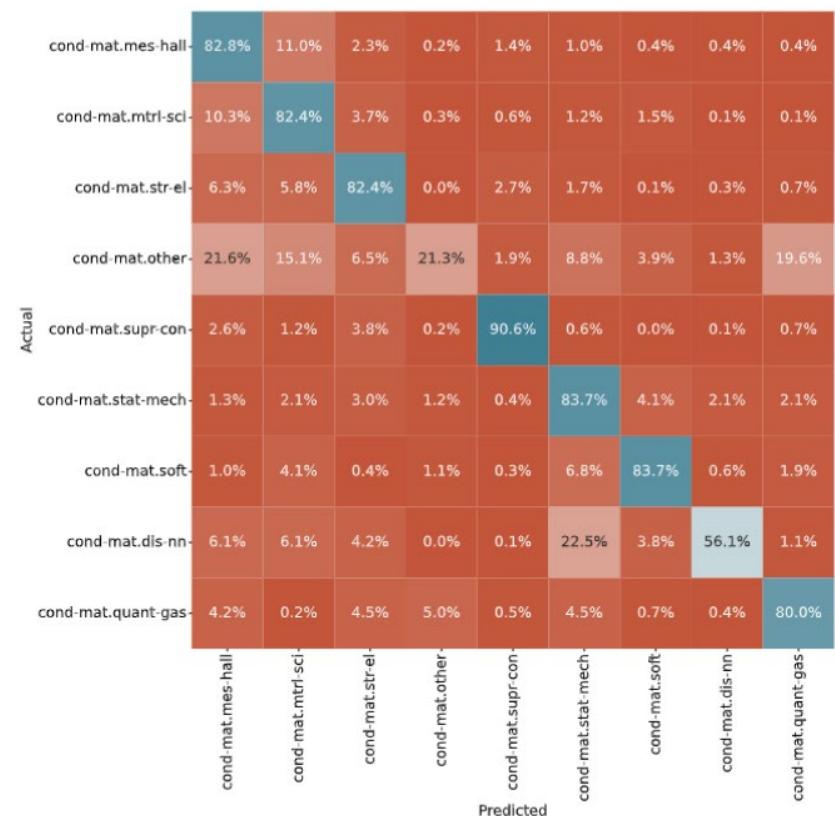
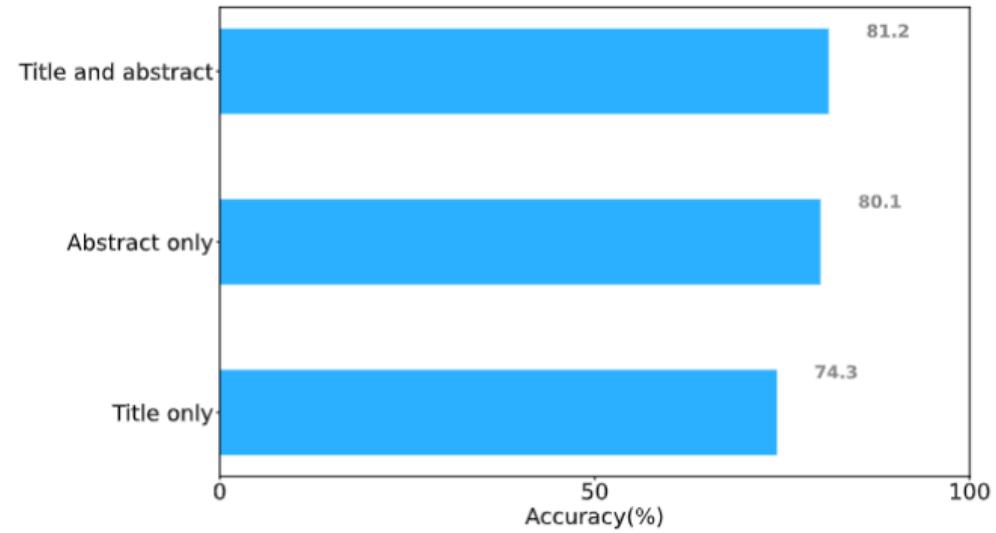
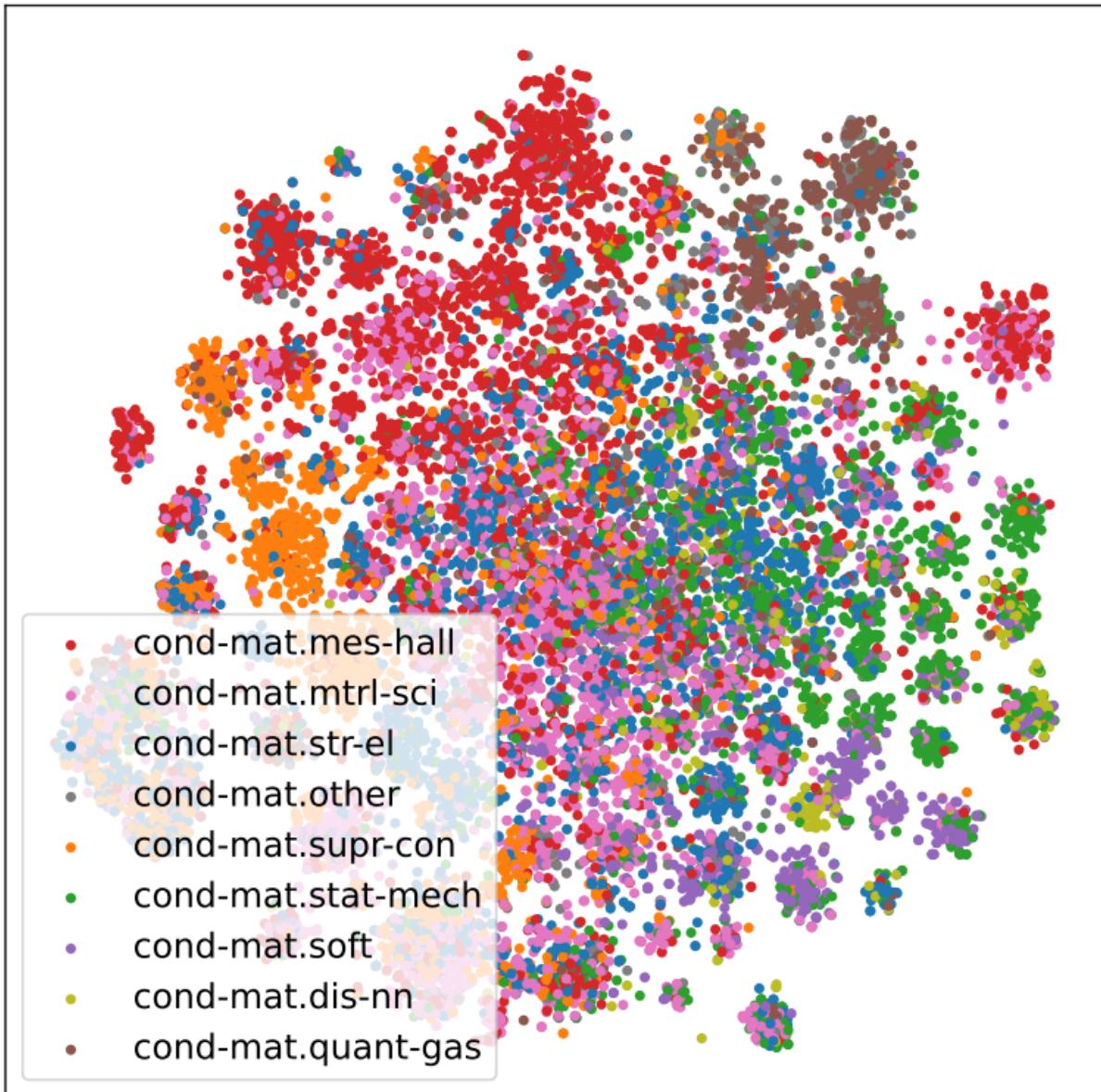


Word-cloud



Venn diagram for chemical formula available in arXiv cond-mat.supr-con and JARVIS-SuperconDB

ChemNLP



JARVIS-Leaderboard

← → ⌂ pages.nist.gov/jarvis_leaderboard/



≡ JARVIS-Leaderboard

Search

jarvis_leaderboard
v2023.03.03 ★ 7 ♫ 1

JARVIS Leaderboard

This project provides benchmark-performances of various methods for materials science applications using the datasets available in JARVIS-Tools databases. Some of the methods are: Artificial Intelligence (AI), Electronic Structure (ES), Quantum Computation (QC) and Experiments (EXP). There are a variety of properties included in the benchmark. In addition to prediction results, we attempt to capture the underlying software, hardware and instrumental frameworks to enhance reproducibility. This project is a part of the NIST-JARVIS infrastructure.

- Number of benchmarks: 296
- Number of tasks: 202

?

How to use this website?



Summary & future works

- Deep-learning for various data-types
- Many state-of-the-art models achieved with GNNs
- Notebooks and GitHub repos; contributions welcome!
- GNN and other methods for benchmarking

Slides: <https://www.slideshare.net/KAMALCHOUDHARY4>

Email: kamal.choudhary@nist.gov



 @dr_k_choudhary

 GitHub @knc6



npj | computational materials

Explore content ▾ About the journal ▾ Publish with us ▾

nature > npj computational materials > articles > article

Article | Open Access | Published: 12 November 2020

The joint automated repository for various integrated simulations (JARVIS) for data-driven materials design

Kamal Choudhary , Kevin F. Garrity, [...] Francesca Tavazza

npj Computational Materials **6**, Article number: 173 (2020) | Cite this article

<https://jarvis.nist.gov>

<https://github.com/usnistgov/jarvis>

<https://github.com/usnistgov/alignn>

<https://github.com/usnistgov/atomvision>

<https://github.com/usnistgov/chemnlp>