Data science and machine learning for material science

Saaketh Desai, Juan Carlos Verduzco, Ale Strachan

desai61@purdue.edu

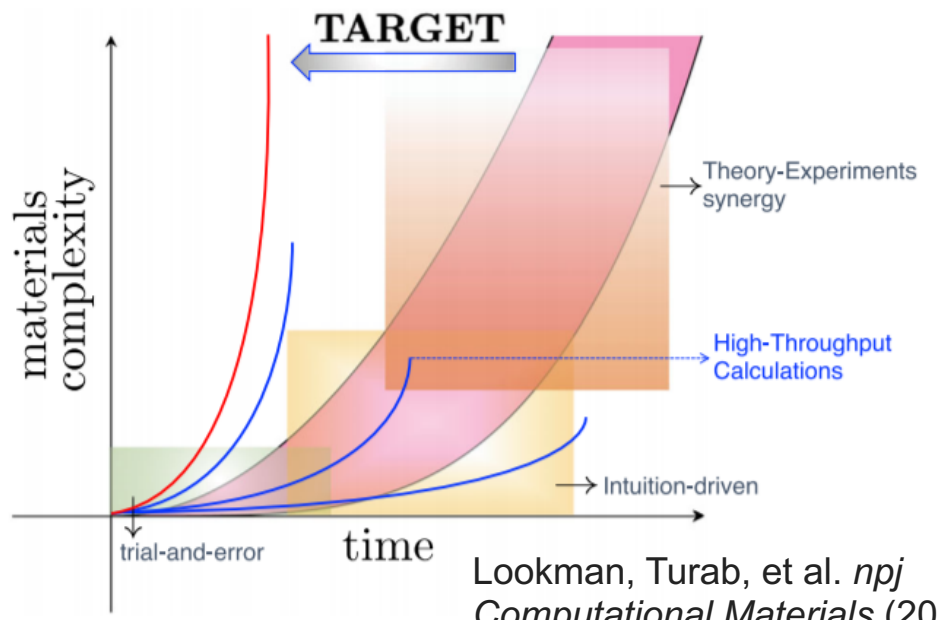
jverduzc@purdue.edu

strachan@purdue.edu

School of Materials Engineering and Birck nanotechnology center

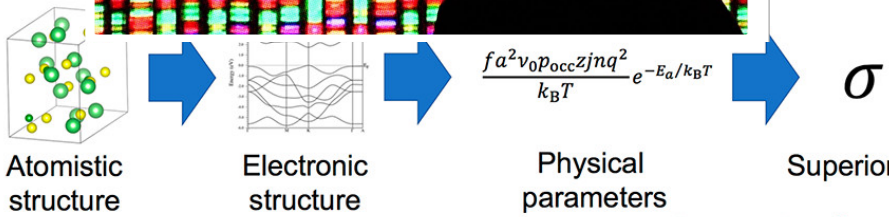
Purdue University

Materials by design



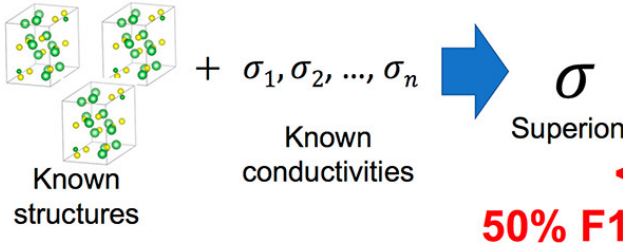
Lookman, Turab, et al. *npj Computational Materials* (2019)

Conve



~4 weeks/prediction

Our machine learning approach



Sendek, Austin D., et al. *Chemistry of Materials* (2018)

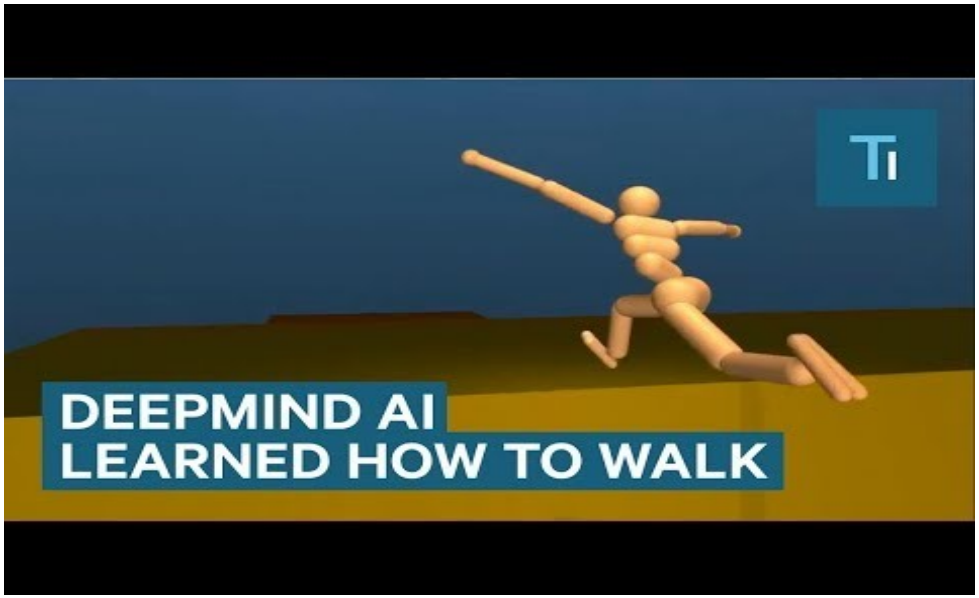
<1 second/prediction

50% F1 score to DFT results

Ferrium® C64®

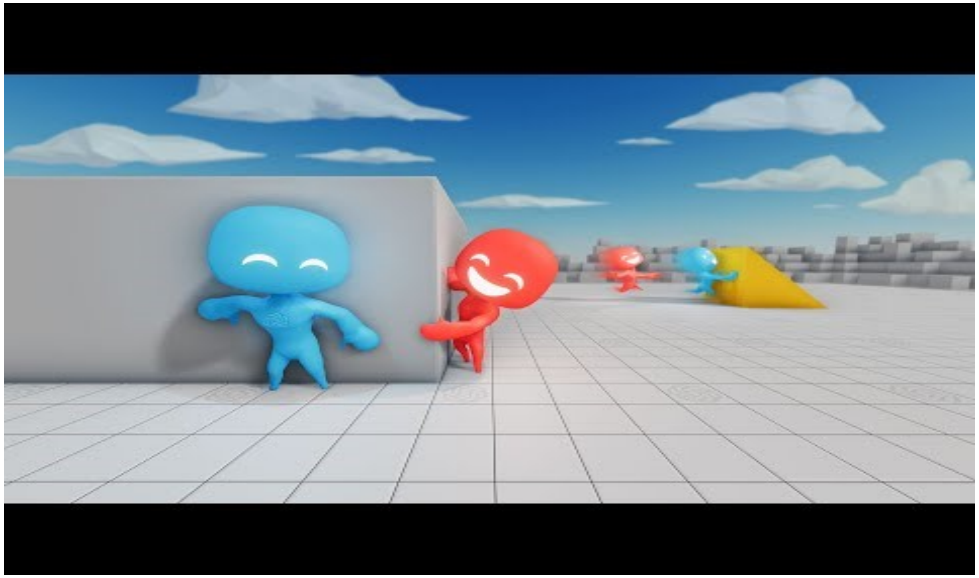
https://www.questek.com/ferrium-c64.html

Data science, machine learning and AI

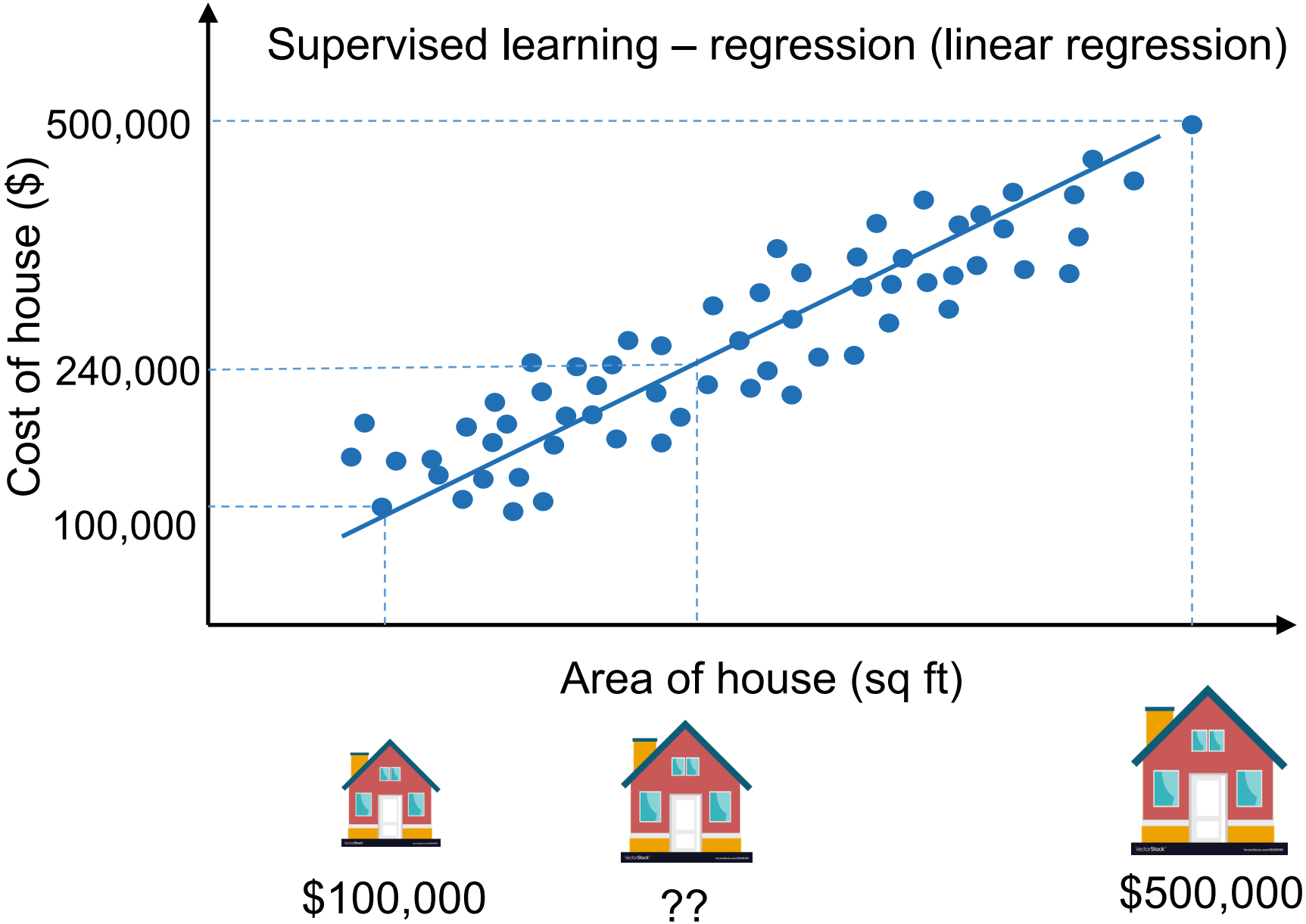


amazon

NETFLIX



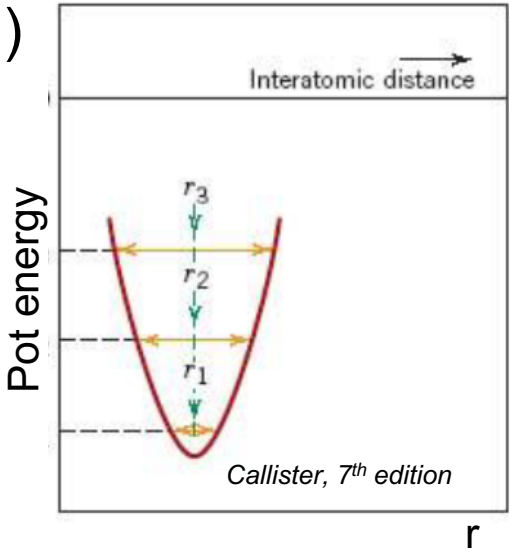
Machine learning and data science



Linear regression to predict Young's modulus

Supervised learning – regression (linear regression)

- **Supervised learning:** Learning input-output relations based on a set of labeled 'ground truth' data
- **Regression:** Predicting a continuous relationship between inputs and outputs
- **Linear regression:** Assumed relationship to be linear



Can we predict Young's modulus based on the melting temperature?

Introduction to Machine Learning for Materials Science

The tutorials here will give you an insight into the usage of Machine Learning to approach problems related to materials science.

- **Get started** Click on the links below to begin each tutorial.
- **Important** To exit individual tutorials and return to this page, use File -> Close and Halt. "Terminate Session" (top right) will kill your entire Jupyter session.

Querying databases, Organizing and Plotting Data:

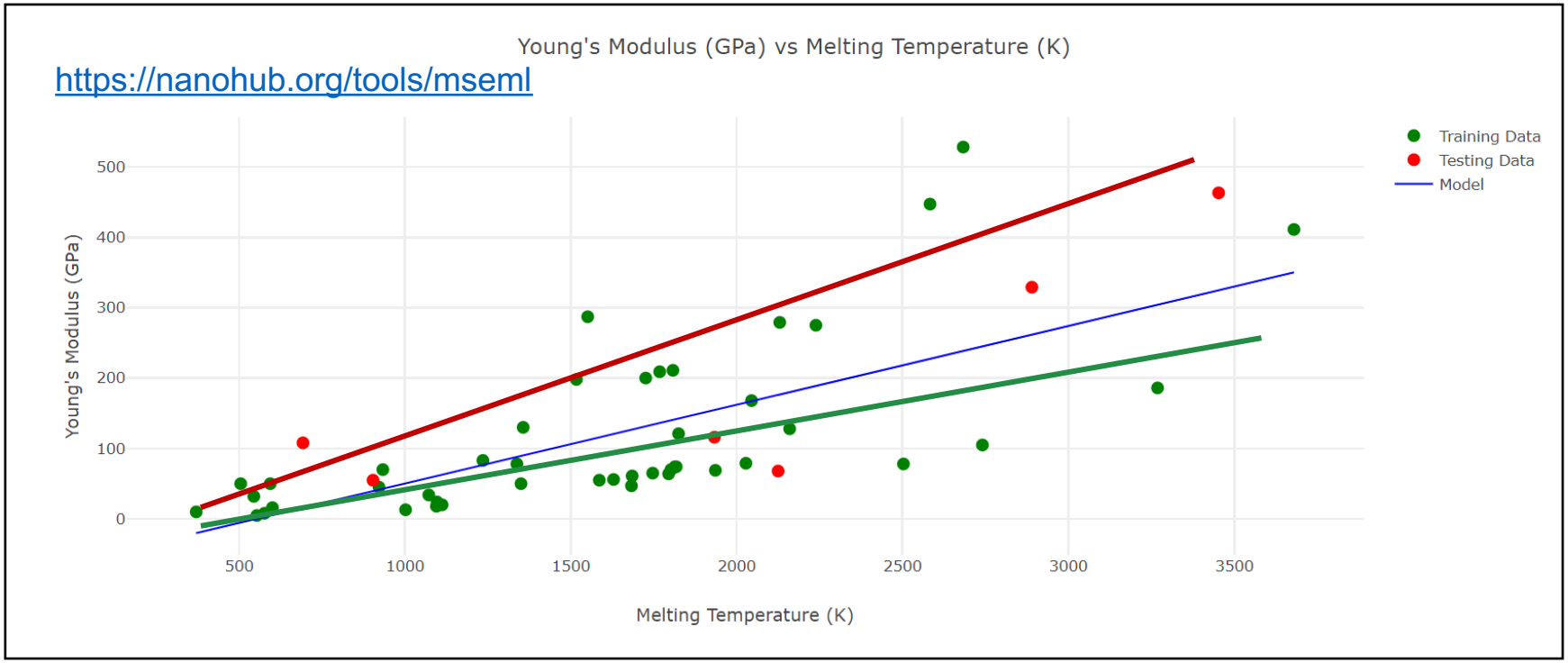
- Query Pymatgen and Mendeleev for properties like Young's modulus and melting temperature
- Organize data into Pandas dataframes and python dictionaries and plot using Plotly

Linear Regression to predict material properties:

- Perform linear regression using the scikit learn package and predict Young's modulus
- Visualize trends in data and 'goodness of fit' of linear model

https://nanohub.org/tools/mseml

Obtaining good models – objective functions & gradient descent



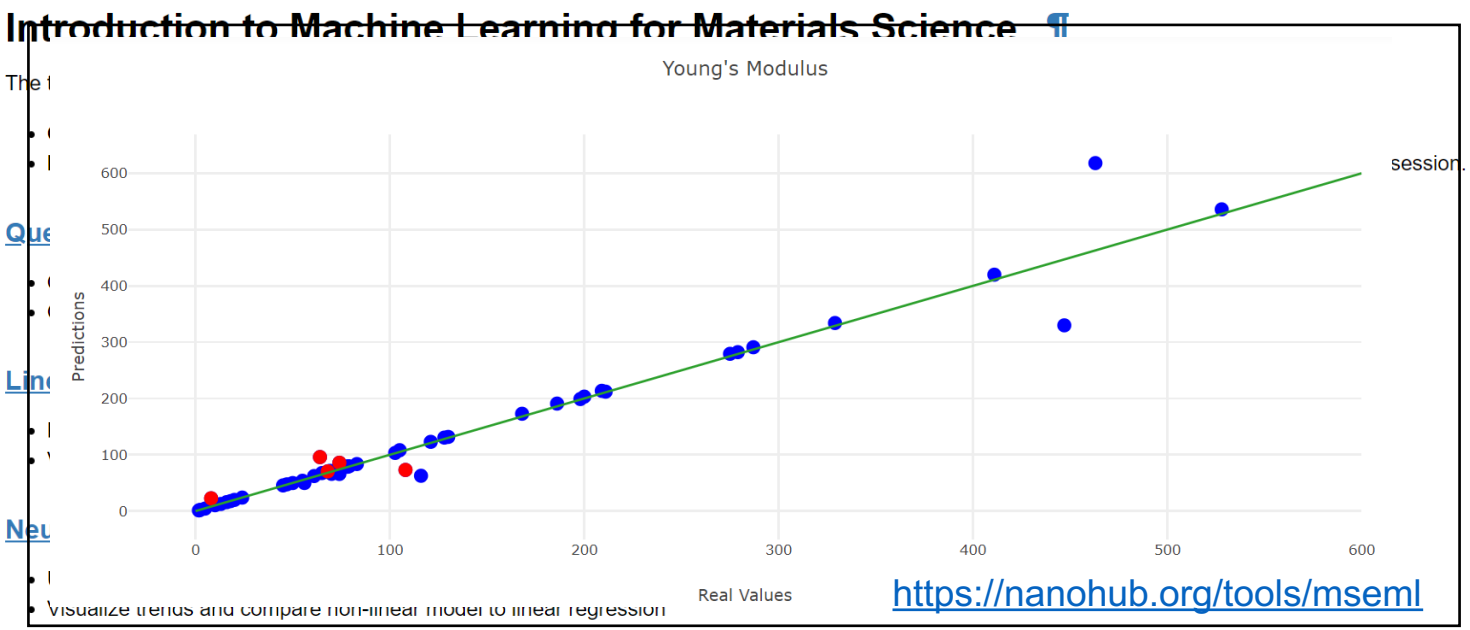
- Objective function: metric to assess model quality
- Algorithm to direct model towards objective: Gradient descent

- Objective function: minimize sum of squared distances (points should lie as close to line as possible)
- Algorithm: move in the direction of largest decrease in error

Non-linear models: Neural networks

We could predict the Young's modulus given the melting temperature
Can we do better?

heat_of_formation	lattice_constant	melting_point	specific_heat	atomic_mass	atomic_radius	electrical_resistivity
284.9	4.09	1235.10	0.237	107.868200	1.60	1.630000e-08
330.9	4.05	933.50	0.900	26.981539	1.25	2.700000e-08
368.2	4.08	1337.58	0.129	196.966569	1.35	2.200000e-08
337.4	3.61	1356.60	0.385	63.546000	1.35	1.720000e-08
669.0	3.84	2683.00	0.133	192.217000	1.35	4.700000e-08



https://nanohub.org/tools/mseml

Neural networks – Forward propagation

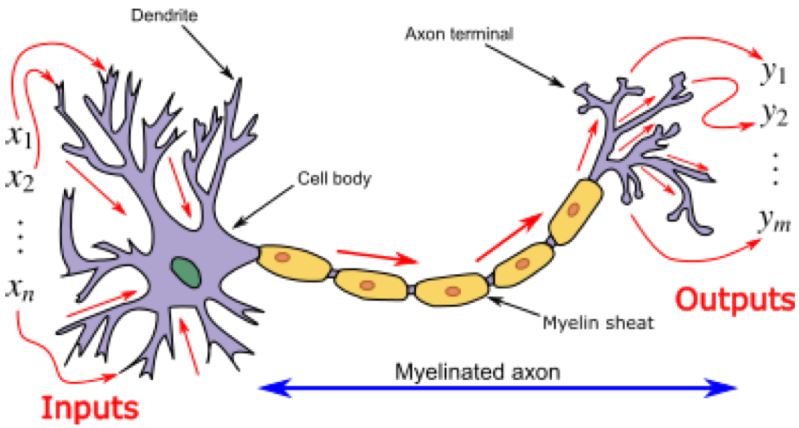
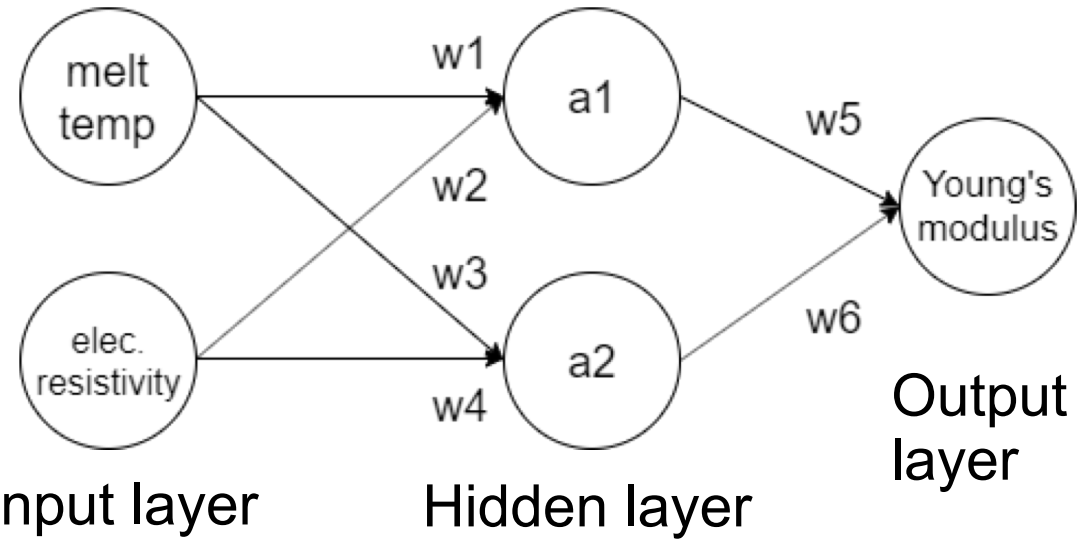


Photo Credit: TowardsDataScience



$$a_1 = f_1(w_1 melt.temp + w_2 elec.resistivity + b_1)$$

activation weight bias

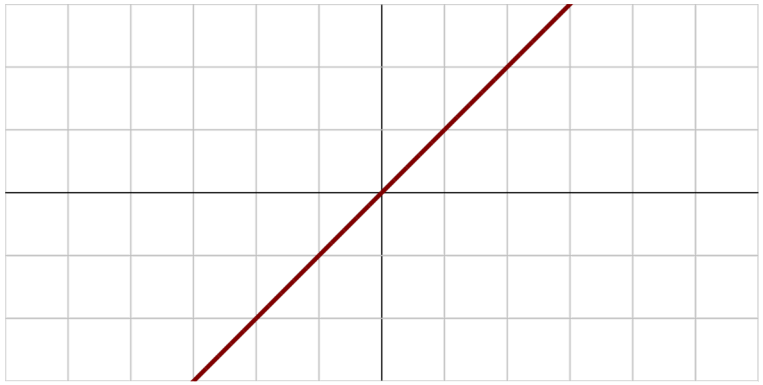
$$a_2 = f_2(w_3 melt.temp + w_4 elec.resistivity + b_2)$$

$$Young's\ modulus = f_3(w_5 a_1 + w_6 a_2 + b_3)$$

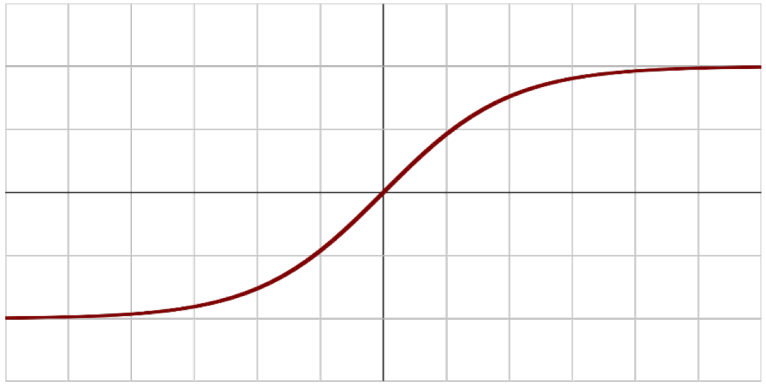
Training a neural network is an optimization problem where we solve for the weights and biases that result in accurate predictions

Neural networks – Activation functions

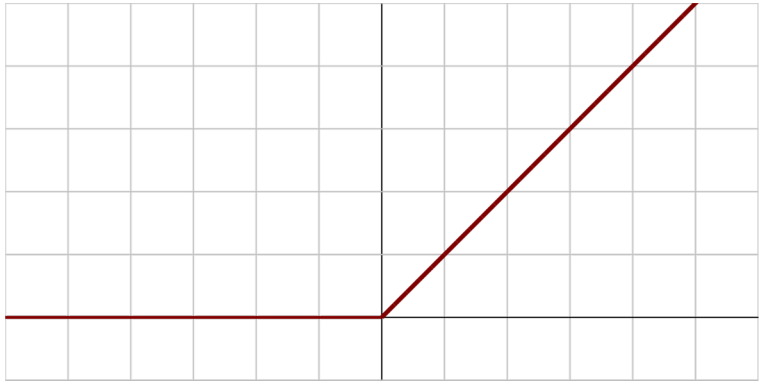
$$Linear$$
$$f(x) = x$$



$$Tanh$$
$$f(x) = \tanh(x)$$



$$Relu$$
$$f(x) = x \; if \; x > 0$$
$$f(x) = 0 \; if \; x < 0$$



Neural networks – back propagation

$$a_1 = f_1(w_1 melt.temp + w_2 elec.resistivity + b_1)$$

activation weight bias

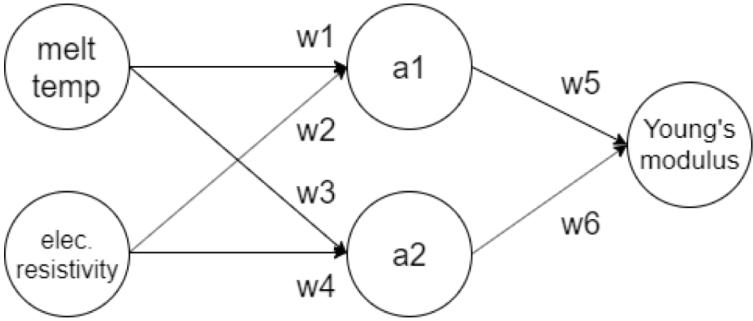
$$a_2 = f_2(w_3 melt.temp + w_4 elec.resistivity + b_2)$$

model prediction

$$\hat{y} = Young's\ modulus = f_3(w_5 a_1 + w_6 a_2 + b_3)$$

$$Objective = \frac{1}{N_{samples}} \sum_{i=1}^{N_{samples}} (\hat{y} - y)^2$$

ground truth



We need to update weights and biases such that objective function is minimized

$$w_1 = w_1 - \alpha \frac{\partial(Objective)}{\partial w_1}$$

Backpropagation

Weight update rule: gradient descent, Adam

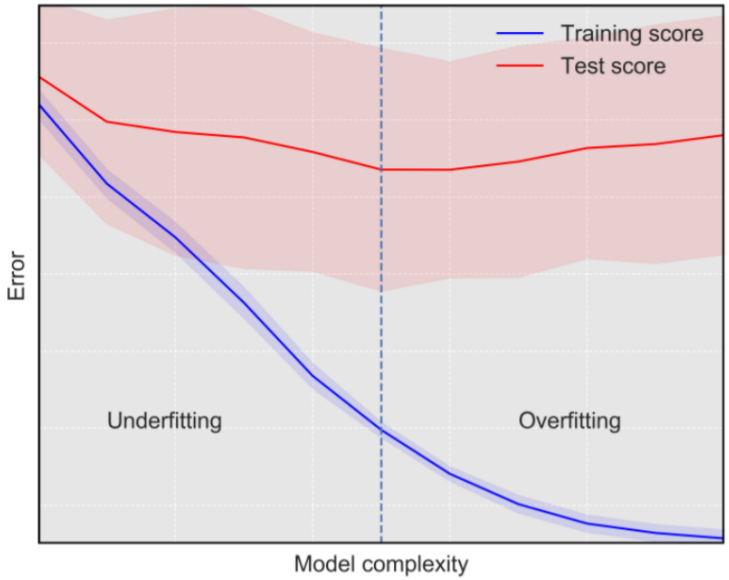
http://neuralnetworksanddeeplearning.com/chap2.html

Underfitting and overfitting

How do we judge if the model has learnt all that it could?

- Underfitting – model hasn't learnt all the trends in the training data
- Overfitting – model has "memorized" data, ignoring the underlying trend

Image: Julien Despois

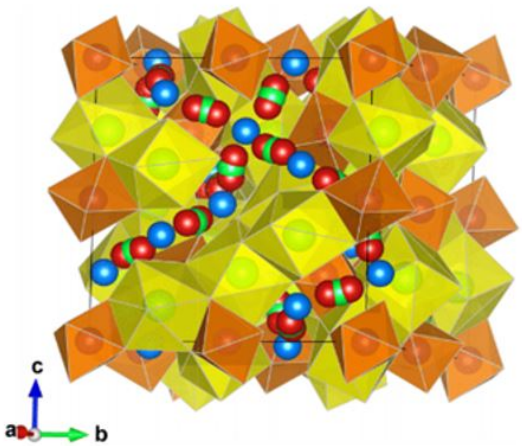


How do we train models that generalize well?

- Use a low enough learning rate
- Monitor error on validation set as a measure of model's ability to generalize

Sequential learning with the Citrination API

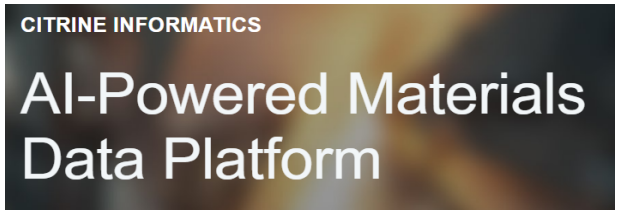
https://nanohub.org/tools/mldev Juan Carlos Verduzco



Rettenwander, Daniel, et al.
Chemistry of materials 26.8 (2014)

threads	natoms	R
1	42592	22
2	85184	27.71826
3	127776	31.72949
4	170368	34.92282
5	212960	37.61947
6	255552	39.97665
7	298144	42.08449
8	340736	44
9	383328	45.76184
10	425920	47.39756
11	468512	48.92756
12	511104	50.36743
13	553696	51.72936
14	596288	53.02313
15	638880	54.25667
16	681472	55.43653
17	724064	56.56819
18	766656	57.65631
19	809248	58.70484
20	851840	59.71719
21	894432	60.69633
22	937024	61.64487
23	979616	62.56507
24	1022208	63.45898
25	1064800	64.32839
26	1107392	65.17491
27	1149984	66
28	1192576	66.80496
29	1235168	67.59097
30	1277760	68.35912

8727.273	threads	natoms	R
	1	8727.273	14.27532
	2	17454.55	17.98577
	3	26181.82	20.58857
	4	34909.09	22.66065
	5	43636.36	24.41045
	6	52363.64	25.93997
	7	61090.91	27.3077
	8	69818.18	28.55063
	9	78545.45	29.69386
	10	87272.73	30.75524
	11	96000	31.74802
	12	104727.3	32.68232
	13	113454.5	33.56605
	14	122181.8	34.40555
	15	130909.1	35.20596
	16	139636.4	35.97155
	17	148363.6	36.70586
	18	157090.9	37.41191
	19	165818.2	38.09228
	20	174545.5	38.74917
	21	183272.7	39.38452
	22	192000	40
	23	200727.3	40.5971
	24	209454.5	41.17714
	25	218181.8	41.74128
	26	226909.1	42.29057
	27	235636.4	42.82595
	28	244363.6	43.34827
	29	253090.9	43.8583
	30	261818.2	44.35673



https://citrination.com/users/sign_in

1. Querying a Database from Citrination

Matminer offers API tools to facilitate querying of databases like the Materials Project and Citrination. An individual **Citrine Key** is requiered for the query command *CitrineDataRetrieval*.

Data is stored in a Pandas Dataframe and the list of possible properties to be queried can be consulted by setting the print_properties_options parameter to **True**.

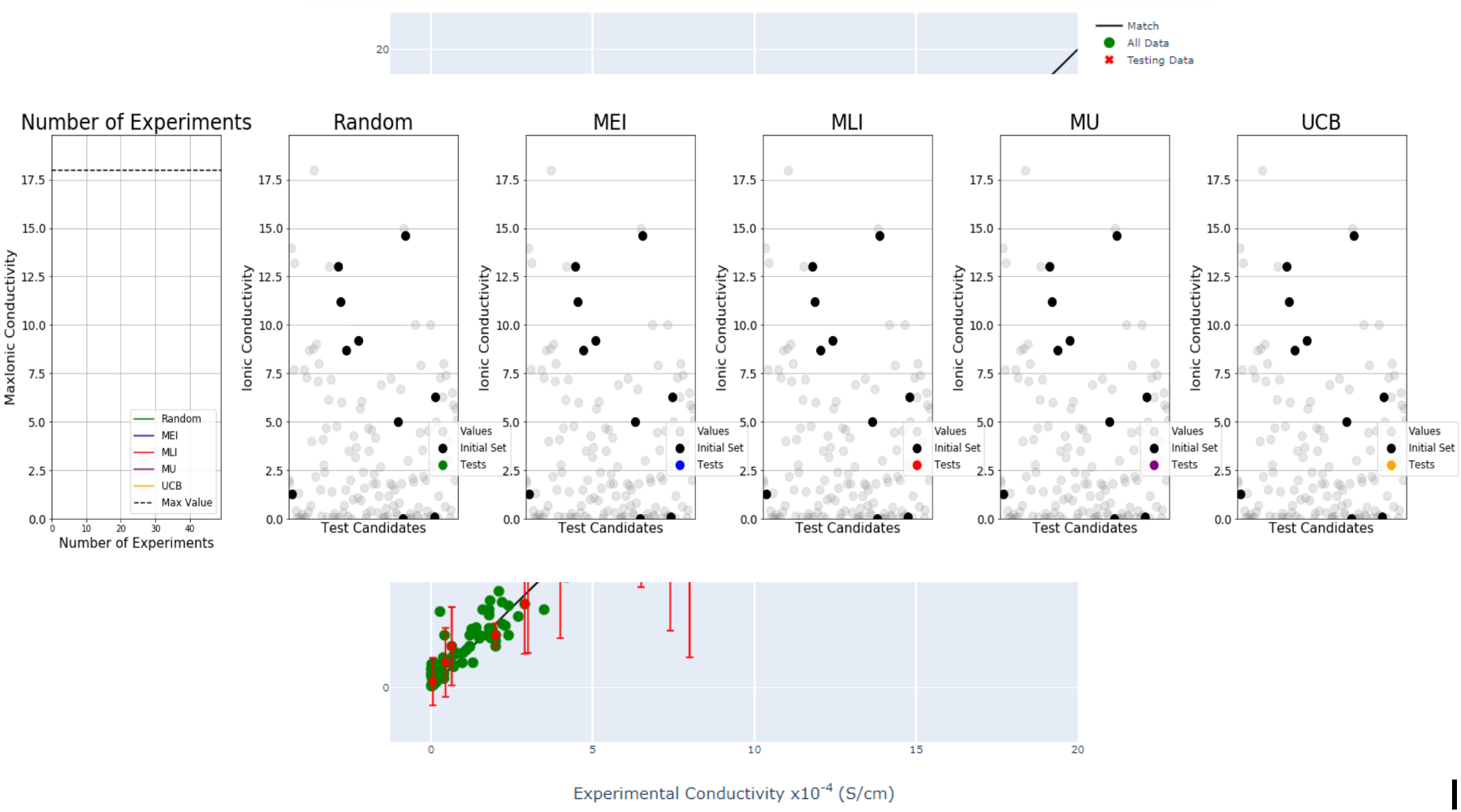
In [2]:
```
cdr = CitrineDataRetrieval('833skPqkMNhjmqOxZA9ADwtt') # Citrine Key

data = cdr.get_dataframe(criteria={'data_set_id': 184812}, print_properties_options=False) # LLZO Database
property_interest = 'Ionic Conductivity' # Property to be queried

display(data.head(n=10))
```

Consider making data publicly available to data platforms

Sequential learning with the Citrination API

https://nanohub.org/tools/mldev Juan Carlos Verduzco

Sequential learning accelerates identification of high ionic conductivity garnets



Working with small datasets – other ML techniques

Synthesis of TiO_2 nanoparticles by hydrolysis and peptization of titanium isopropoxide solution

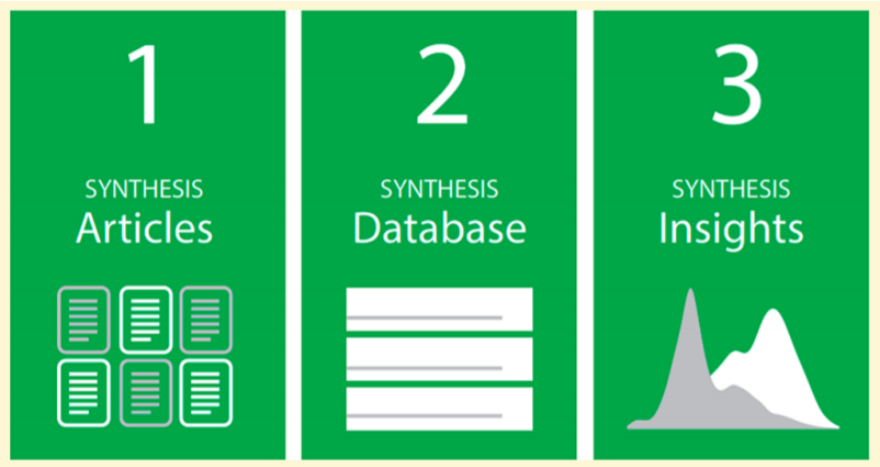
S. Mahshid [a], M. Askari [a], M. Sasani Ghamsari [b,*]

[a] *Department of Material Science & Eng., Sharif Industrial University, 11365-9466 Tehran, Iran*
[b] *Solid State Laser Division, Laser Research Center, North Karegar, 11365-8486 Tehran, Iran*

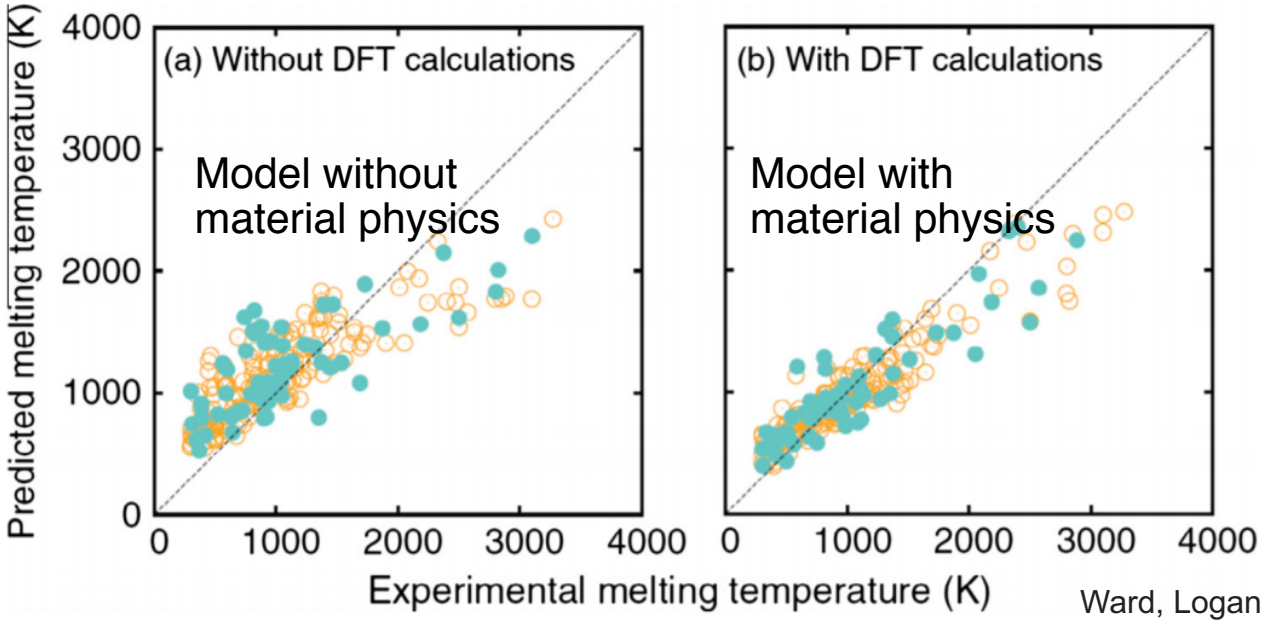
Received 12 July 2006; received in revised form 16 December 2006; accepted 29 January 2007

The prepared precipitates were washed with ethanol and dried for several hours at 100 °C. After being washed with ethanol and dried at 100 °C in a vacuum system for 3 h, a yellow-white powder is obtained. Finally, the prepared powder was annealed at temperature ranging from 200 to 800 °C for 2 h.

ML can automate text mining and information acquisition



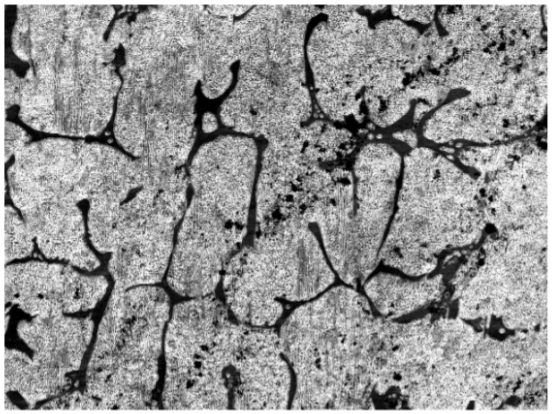
Kim, Edward, et al. *Chemistry of Materials* (2017)

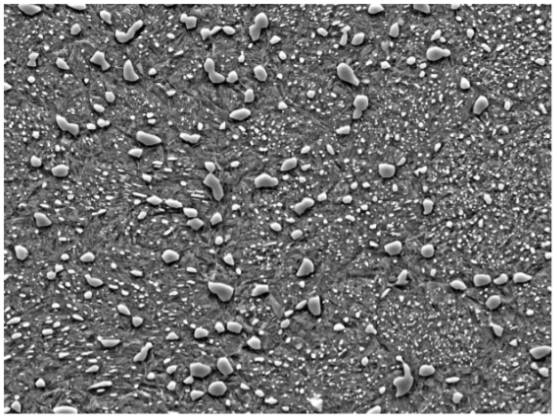


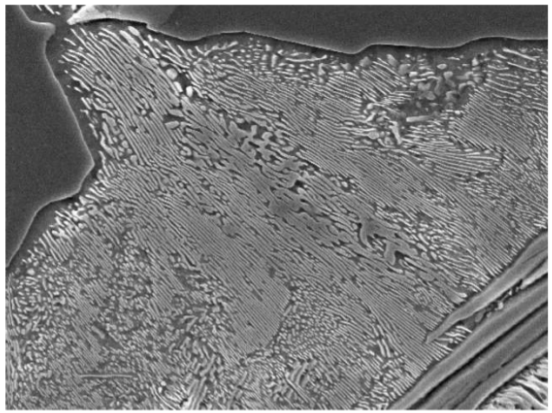
Incorporating physics into the model can significantly improve training even more smaller datasets!

Ward, Logan, and Chris Wolverton, *Current Opinion in Solid State and Materials Science* (2017)

Other applications of neural networks

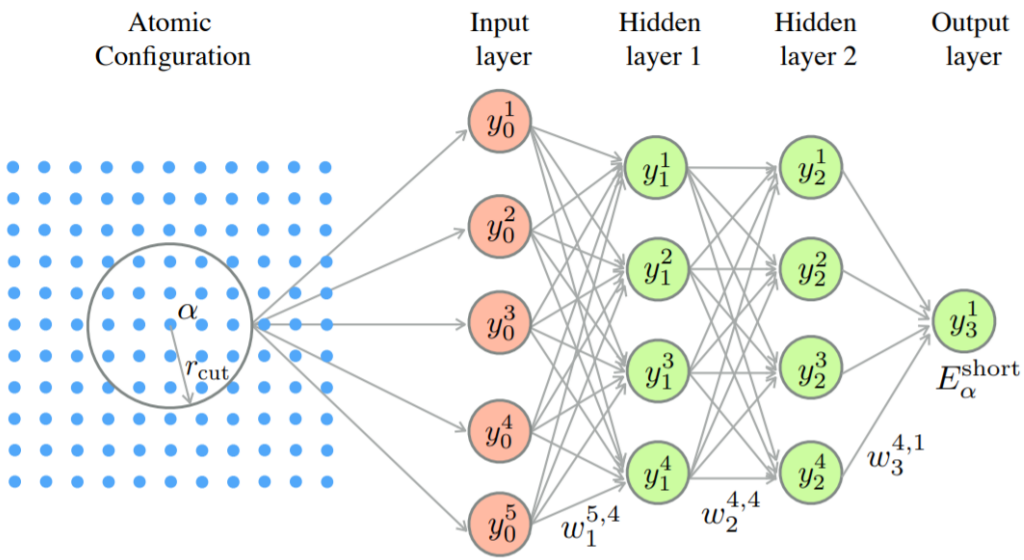






Ling, Julia, et al., *Materials Discovery* 10 (2017): 19-28

Convolutional neural networks can detect microstructural features



Artifical neural network interatomic potentials allow molecular dynamics simulations with almost quantum mechanical accuracy

Wen, Mingjian, and Ellad B. Tadmor, *arXiv preprint arXiv:1909.10134* (2019)

Obtaining and querying data

- Materials project: https://materialsproject.org/
- AFLOW: http://aflowlib.org/
- NIST Materials Data Repository: https://materialsdata.nist.gov/
- Database for renewable energy materials: https://materials.nrel.gov/
- OQMD: http://oqmd.org/

- Citrination: https://citrination.com/datasets
- ICSD: https://icsd.fiz-karlsruhe.de/

Databases with Python APIs:

- Pymatgen: https://pymatgen.org/
- Mendeleev: https://mendeleev.readthedocs.io/en/stable/data.html

- NanoHUB: https://nanohub.org/tools/mseml

Training neural networks

- Keras: https://materialsproject.org/
- Scikit-learn: https://scikit-learn.org/stable/
- PyTorch: https://pytorch.org/

- Matminer: https://hackingmaterials.lbl.gov/matminer/

- NanoHUB: https://nanohub.org/tools/mseml