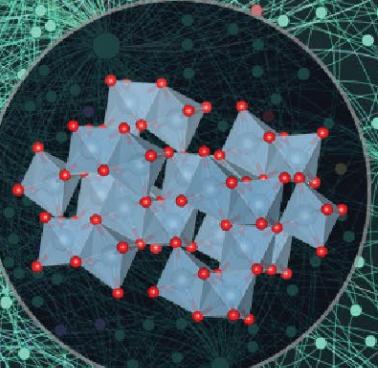
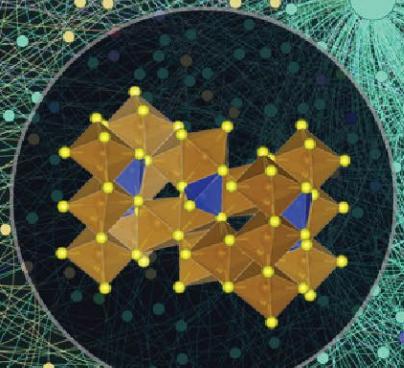
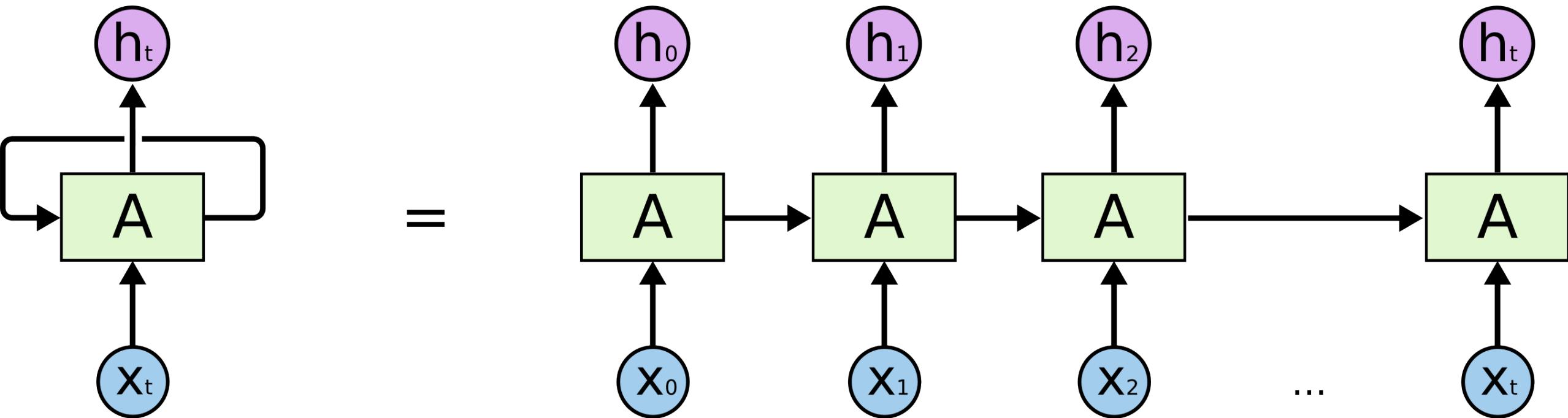


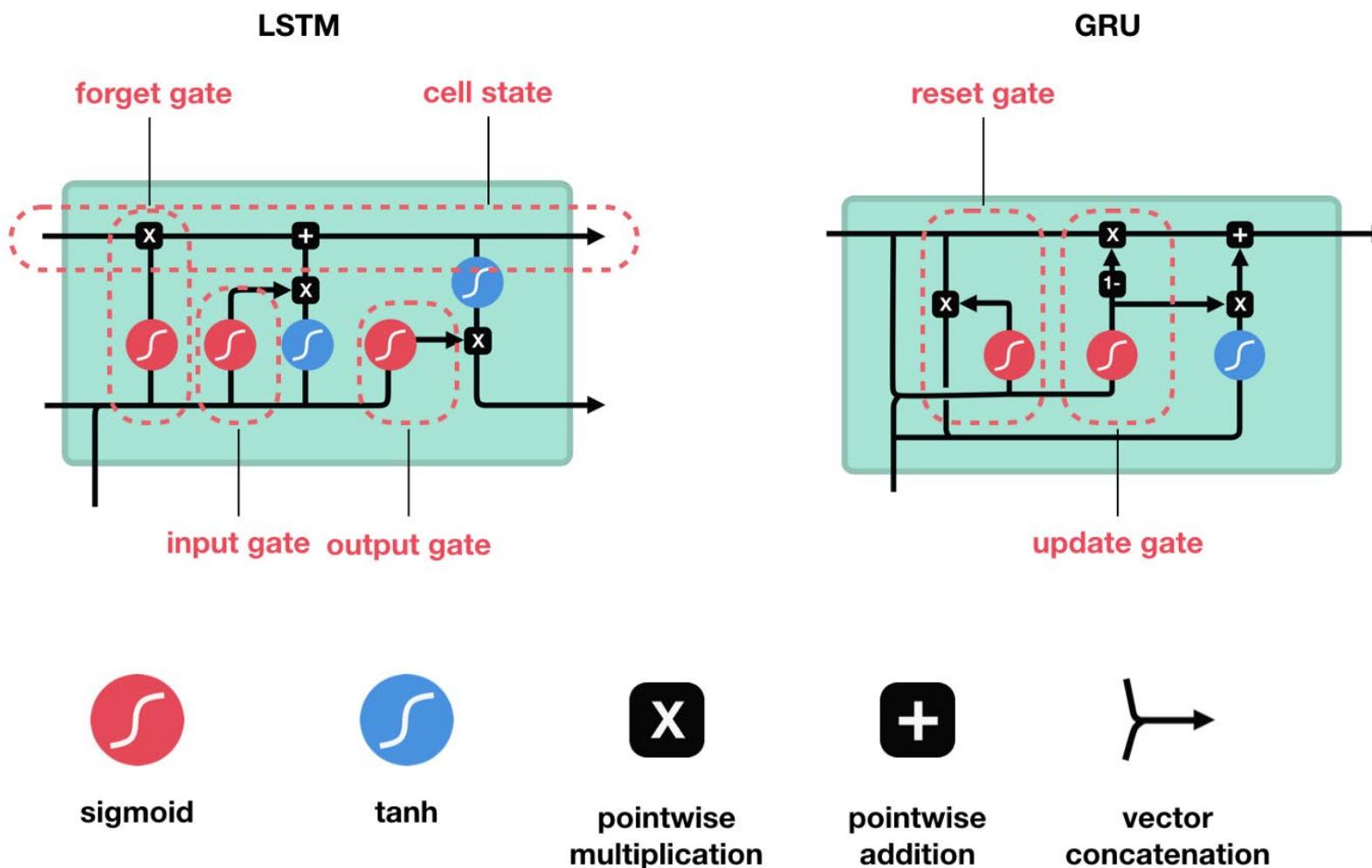
# Transformers



Transformers were created to fix some of the problems of other recurrent neural networks

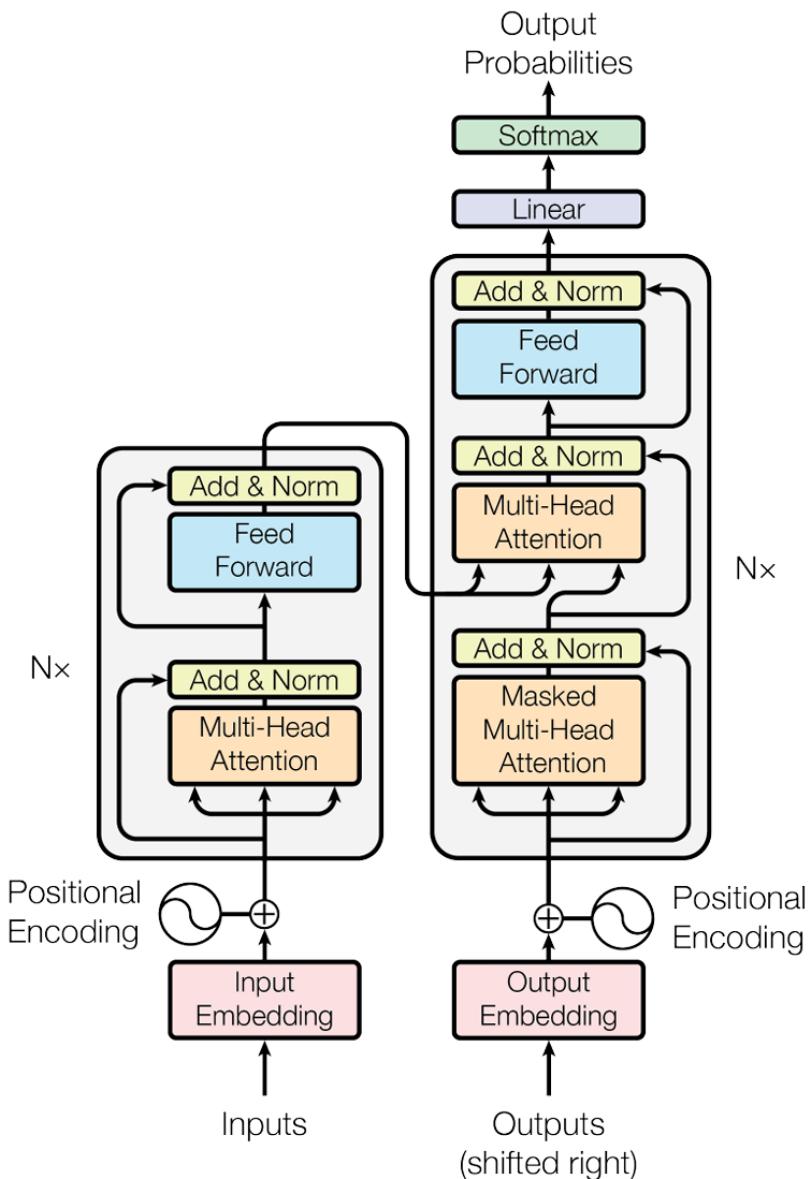


LSTMs were an improvement on RNNs, but they were too slow and still too limited

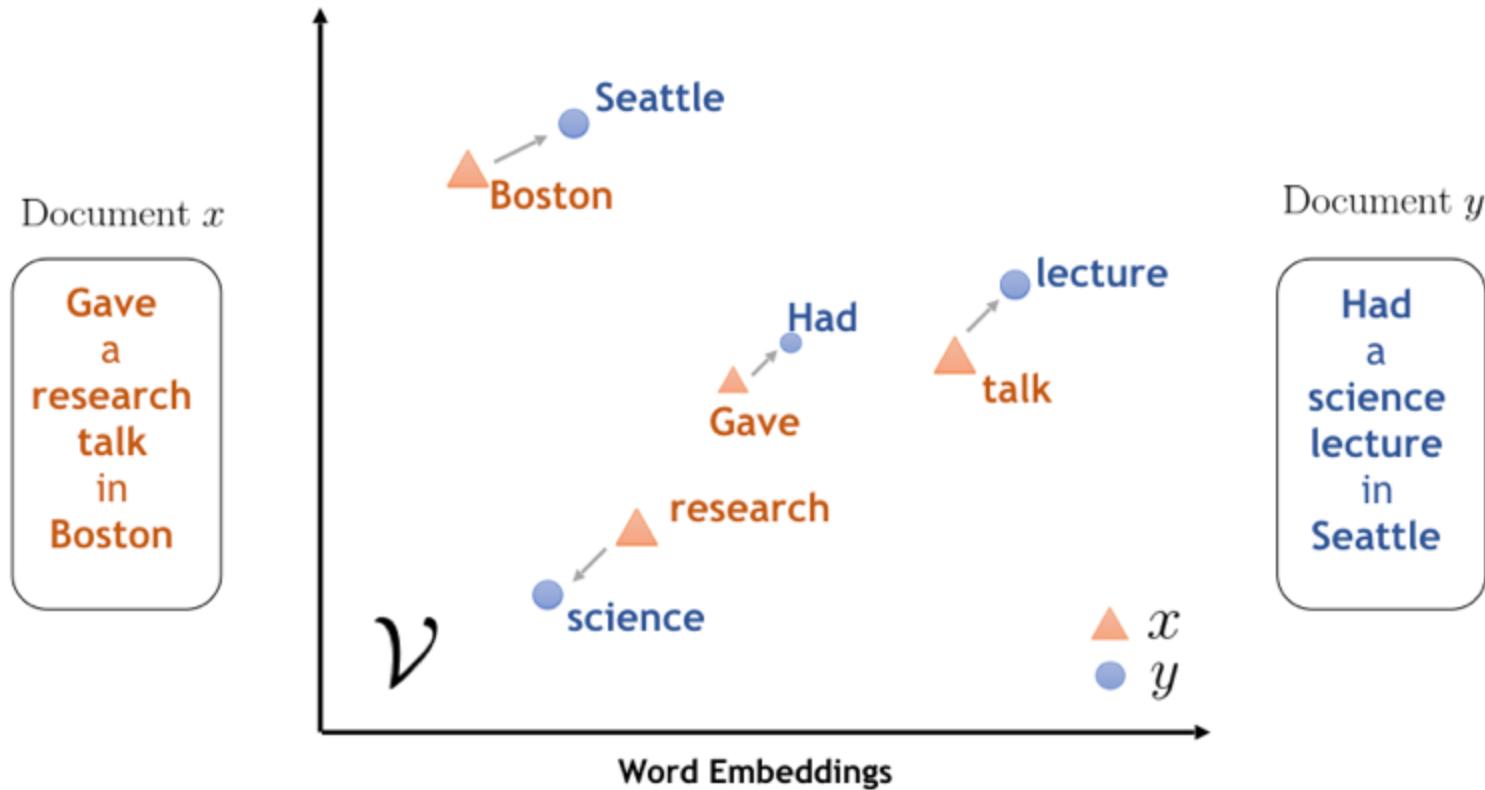


# Transformers allow us to parallelize sequential data!

arXiv:1706.03762v5 [cs.CL] 6 Dec 2017



Transformers first use embeddings to featurize the input\*



\*needed for words, but could be skipped for some inputs

# Embeddings can be generated from training data, or rely on pretrained embeddings

Google Code Archive

Search this site

Projects Search About

**Project**

**word2vec**

Source

Issues

Wikis

Downloads

Tool for computing continuous distributed representations of words.

## Introduction

This tool provides an efficient implementation of the continuous bag-of-words and skip-gram architectures for computing vector representations of words. These representations can be subsequently used in many natural language processing applications and for further research.

## Quick start

- Download the code: svn checkout <http://word2vec.googlecode.com/svn/trunk/>
- Run 'make' to compile word2vec tool
- Run the demo scripts: `./demo-word.sh` and `./demo-phrases.sh`
- For questions about the toolkit, see <http://groups.google.com/group/word2vec-toolkit>

## Project Information

The project was created on Jul 29, 2013.

- License: [Apache License 2.0](#)
- 945 stars
- svn-based source control

Labels:

[NeuralNetwork](#) [MachineLearning](#)  
[NaturalLanguageProcessing](#) [WordVectors](#)  
[Google](#)

# GloVe: Global Vectors for Word Representation

Jeffrey Pennington, Richard Socher, Christopher D. Manning

## Introduction

GloVe is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space.

## Getting started (Code download)

- Download the latest [latest code](#) (licensed under the [Apache License, Version 2.0](#)). Look for "Clone or download"
- Unpack the files: unzip master.zip
- Compile the source: cd GloVe-master && make
- Run the demo script: ./demo.sh
- Consult the included README for further usage details, or ask a [question](#)

## Download pre-trained word vectors

- Pre-trained word vectors. This data is made available under the [Public Domain Dedication and License](#) v1.0 whose full text can be found at: <http://www.opendatacommons.org/licenses/pddl/1.0/>
  - Wikipedia 2014 + Gigaword 5 (6B tokens, 400K vocab, uncased, 50d, 100d, 200d, & 300d vectors, 822 MB download): [glove.6B.zip](#)
  - Common Crawl (42B tokens, 1.9M vocab, uncased, 300d vectors, 1.75 GB download): [glove.42B.300d.zip](#)
  - Common Crawl (840B tokens, 2.2M vocab, cased, 300d vectors, 2.03 GB download): [glove.840B.300d.zip](#)
  - Twitter (2B tweets, 27B tokens, 1.2M vocab, uncased, 25d, 50d, 100d, & 200d vectors, 1.42 GB download): [glove.twitter.27B.zip](#)
- Ruby [script](#) for preprocessing Twitter data

## Citing GloVe

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. [GloVe: Global Vectors for Word Representation](#) [pdf] [bib]

Inputs can have different meanings depending on sequence and context

The deadline creped up on me.

The tensile specimen creped until failure

She conducts the music beautifully.

The ceramic only conducts ions.

I observed necking in my sample.

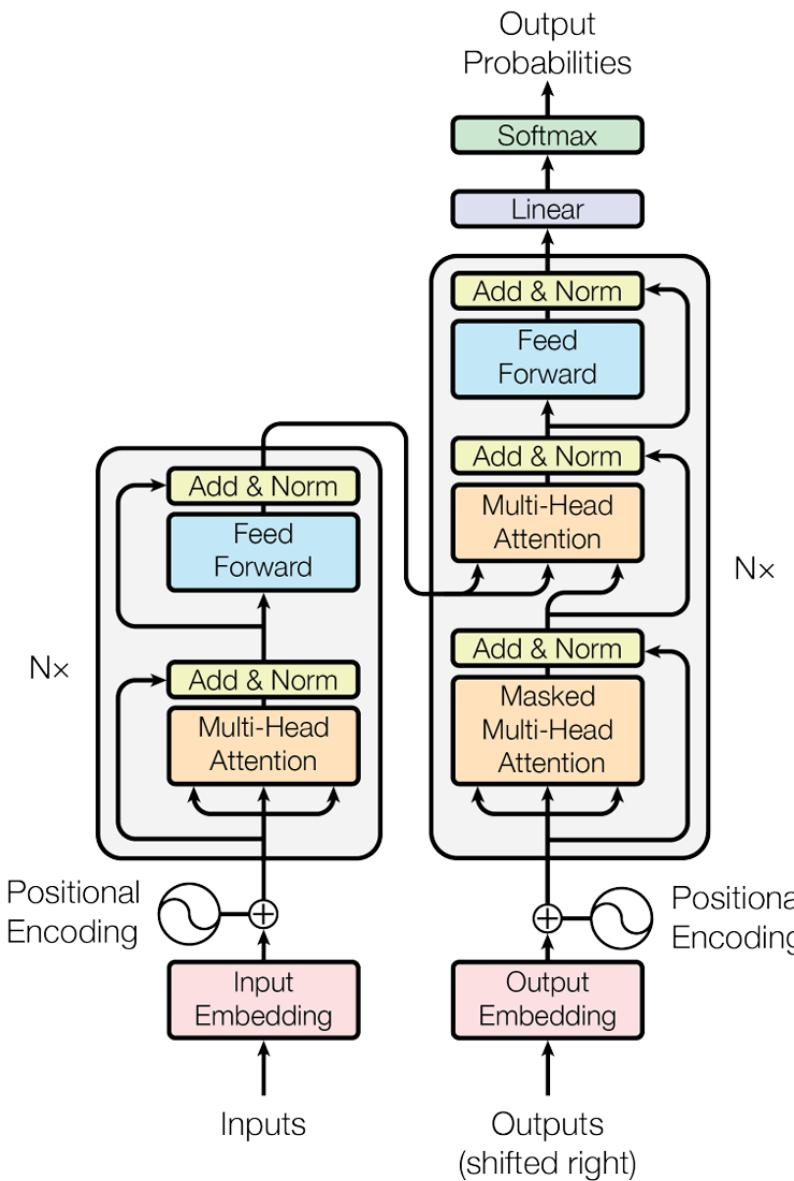
I observed necking in the parked cars at lover's lookout.

To get the right meaning, we need to use a positional encoding

$$\begin{bmatrix} 0.56 \\ 0.77 \\ 0.1 \end{bmatrix} + \textit{positional encoding} = \textit{embedding in context}$$

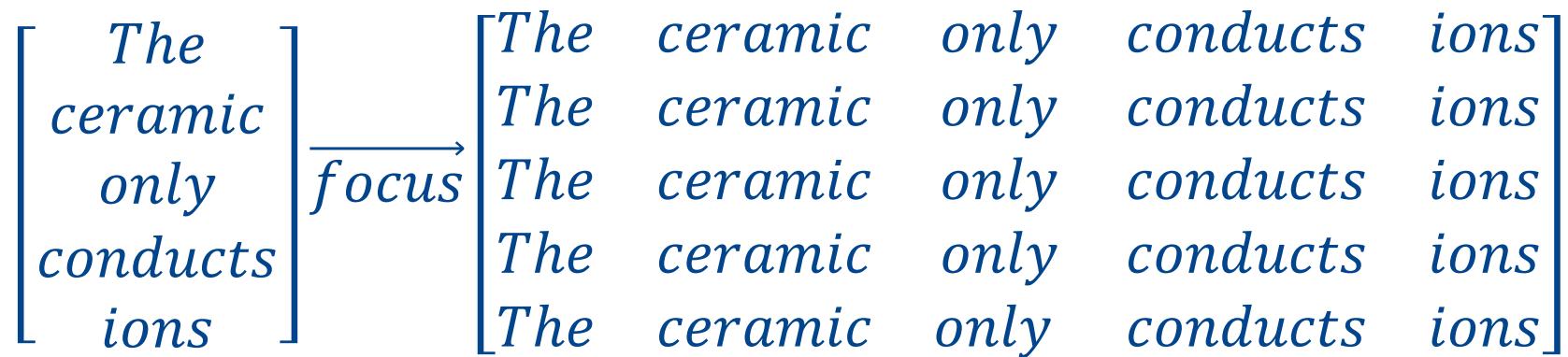
$$PE = \cos(position/1000^{2i/d\_model})$$

Inputs are passed to “encoder block” with multi-head attention and feed forward layers



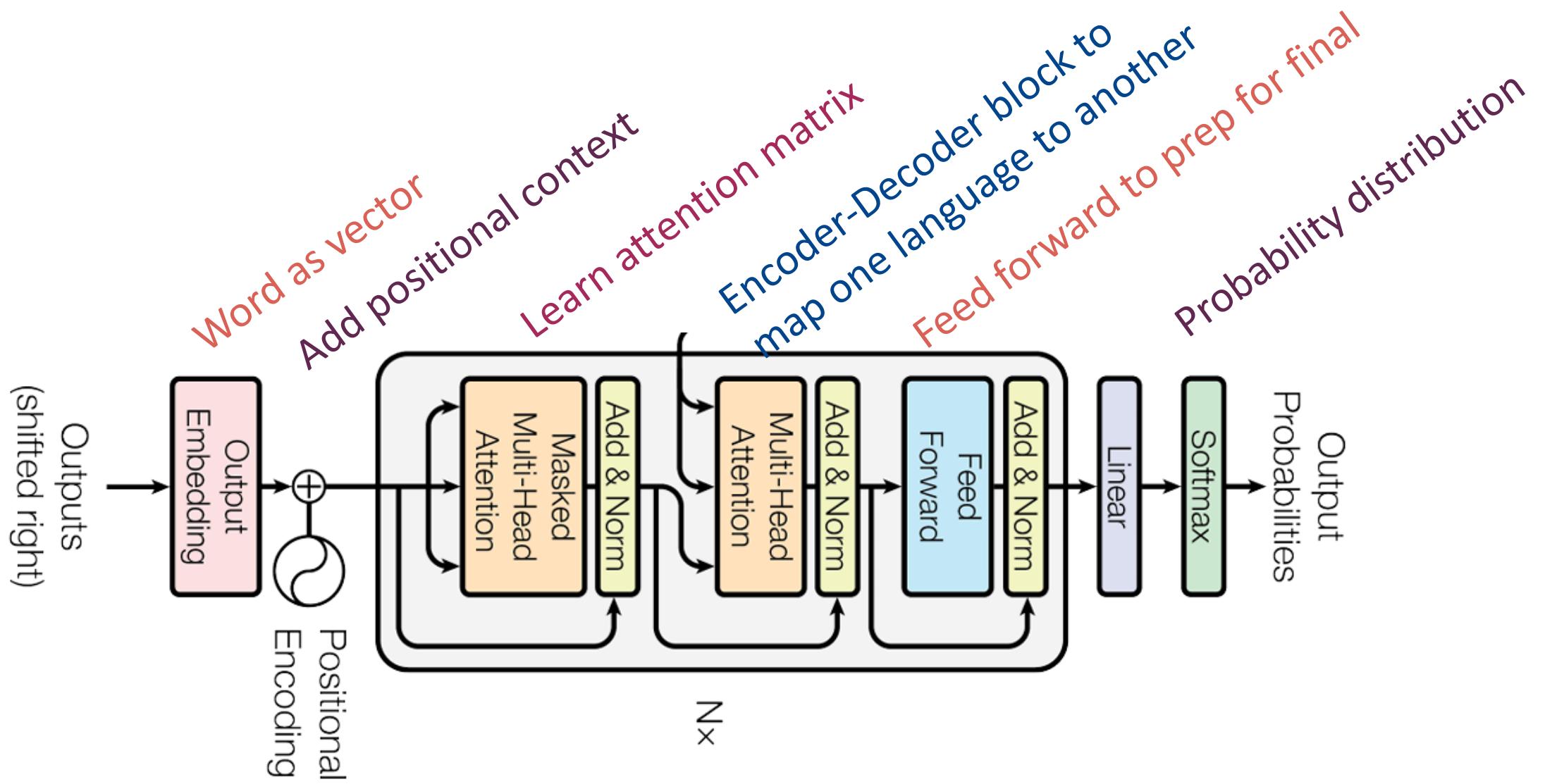
Attention vectors tell us which each input pays attention to in the sequence of data

The ceramic only conducts ions.



These attention vectors for each input are then sent through feed forward layer to transform these vectors into a layer that can be used for next step: “the decoder block”

The decoder block generates output sequence one element at a time based on encoder input



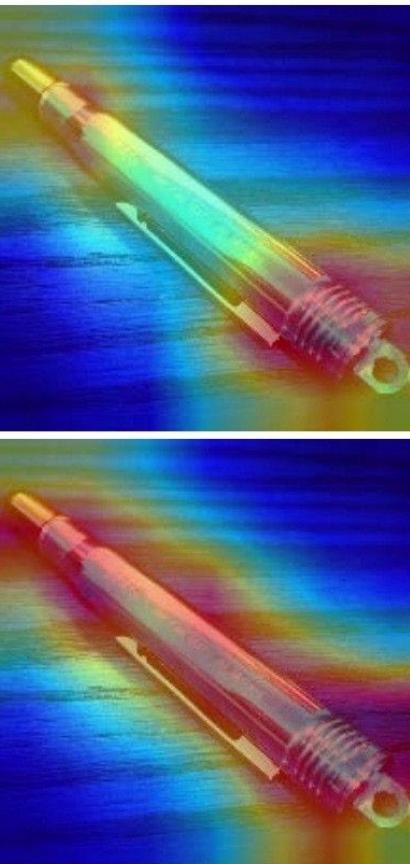
Multi-head attention is used to get multiple perspectives on where to focus



Bulbul



Mountain dog



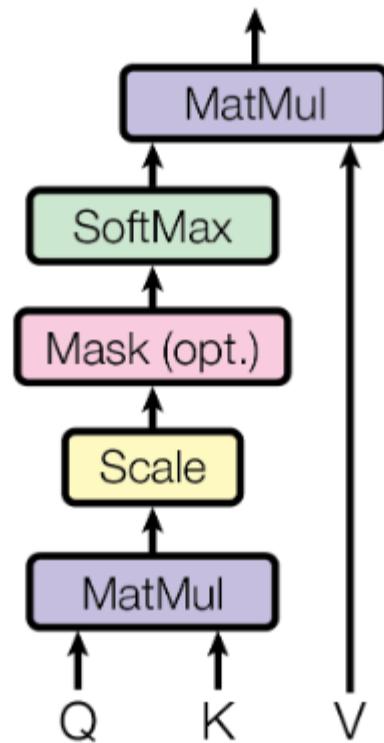
Ballpoint

BONUS: each attention map is done independently from others, so we can parallelize this part of the process!

Encode entire sequence in parallel

Each attention layer contains query, key, and value to learn relationships

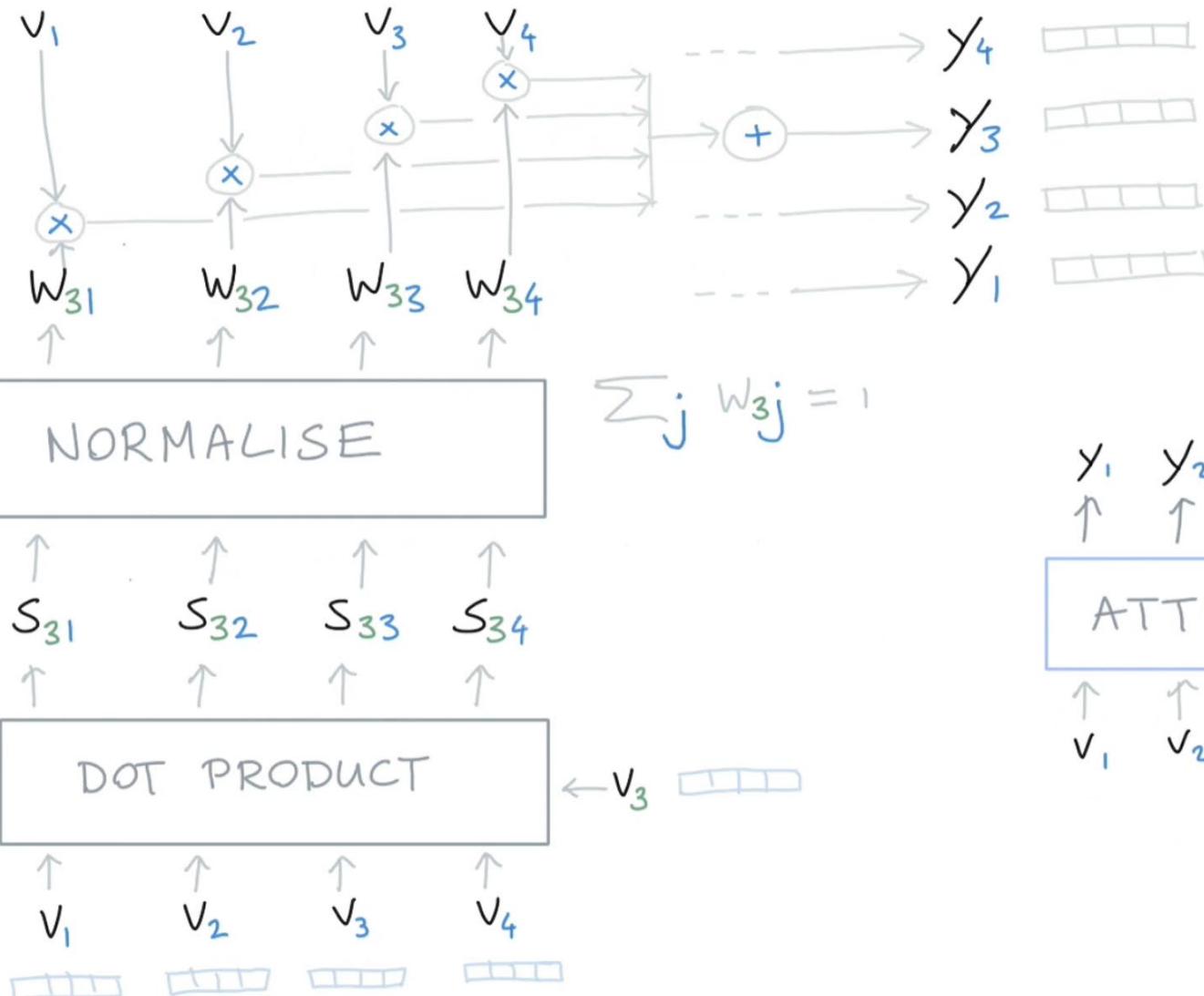
## Scaled Dot-Product Attention



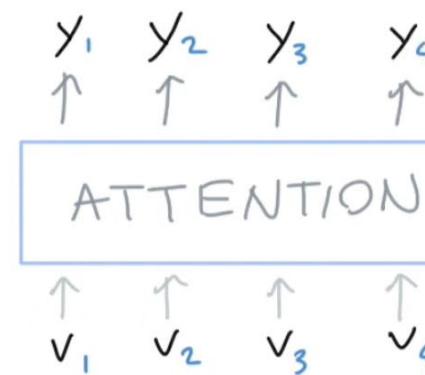
We use  $Q$ ,  $K$ ,  $V$  (abstract vectors) to compute an attention vector for each word

$$Z = \text{softmax} \left( \frac{Q \cdot K^T}{\sqrt{\text{Dimension of vector } Q, K, \text{ or } V}} \right) \cdot V$$

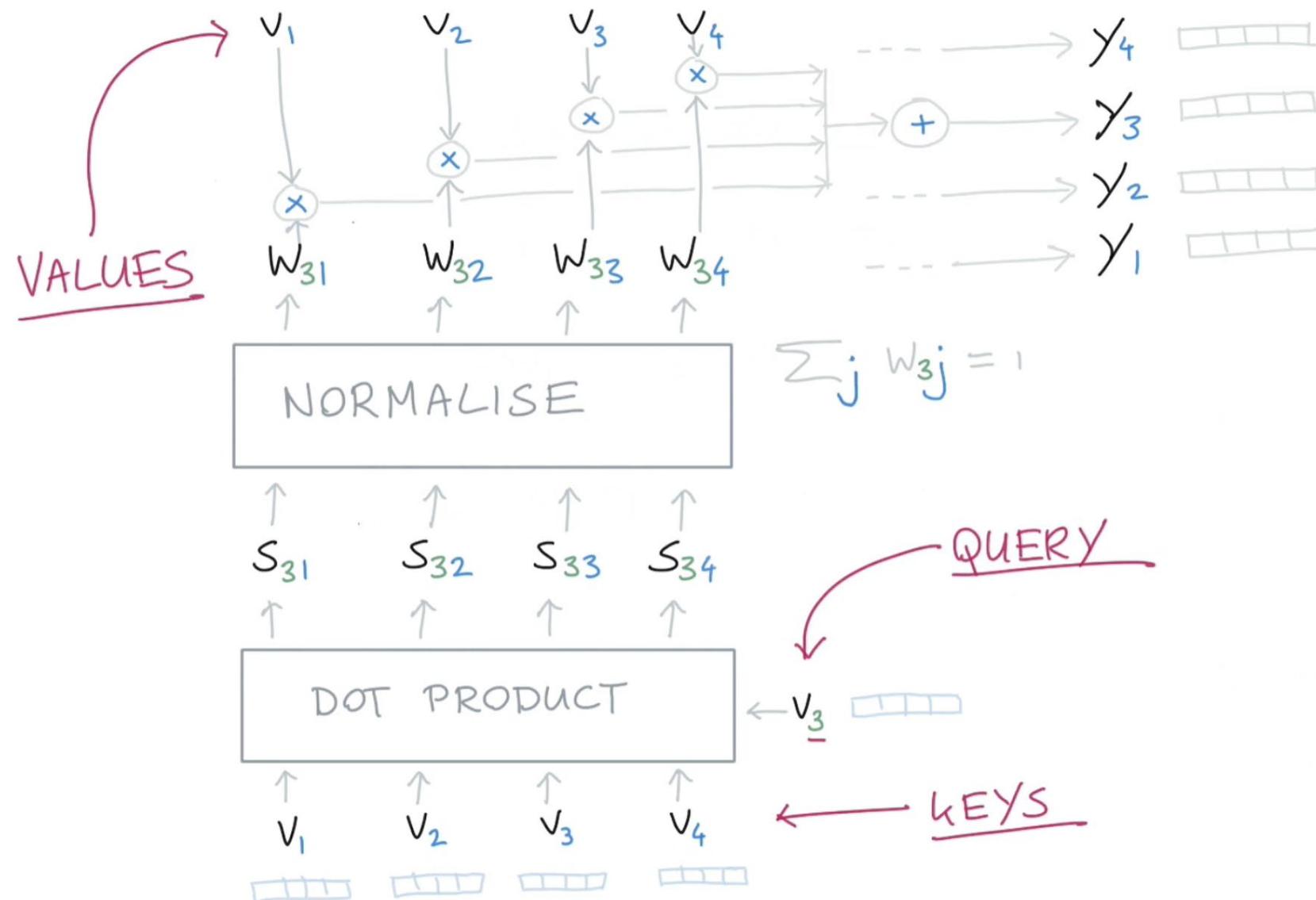
The attention block adds context!



None of this has any tunable parameters to learn during training!



We can add tunable parameters to make context pattern finding more efficient



Keys:  $v_i M_k$   
Queries:  $v_i M_Q$   
Values:  $v_i M_V$

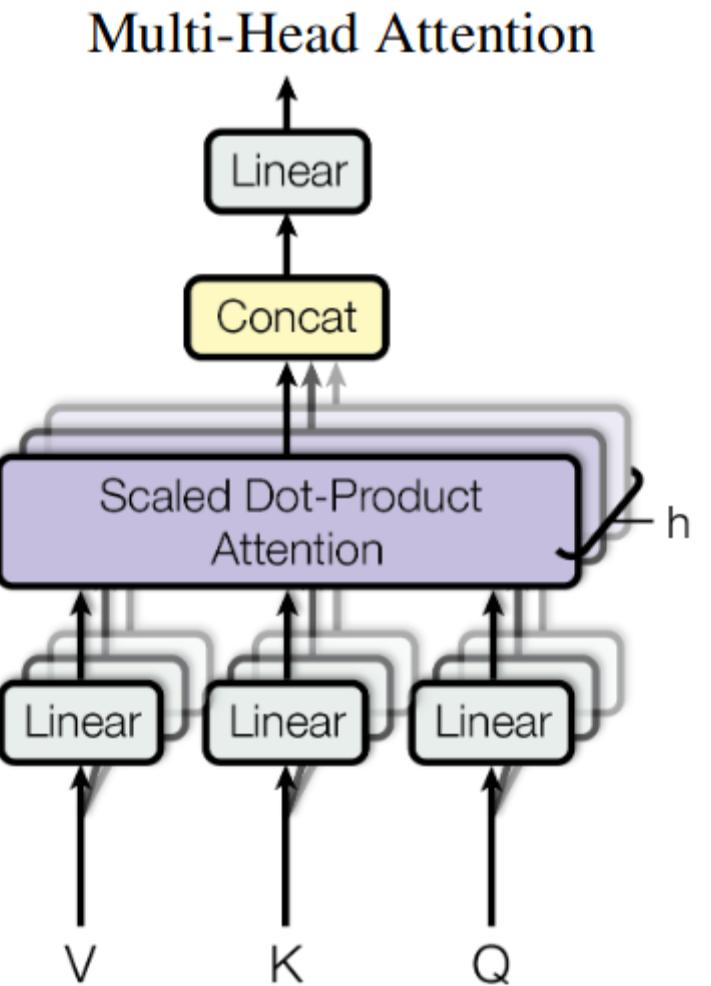
Multi-head attention requires multiple attention vectors for each input

We now have matrices of  $W^V, W^K, W^Q$

$$Z = \text{softmax} \left( \frac{Q \cdot K^T}{\sqrt{\text{Dimension of vector } Q, K, \text{ or } V}} \right) \cdot V$$

Since neural net only expects one vector per word, we flatten to a single vector per word with a weighted matrix

$$Z = [Z_1, Z_2, Z_3 \dots] \cdot W^Z$$



# Tensorflow has a nice code repository for attention networks

TensorFlow    Install    Learn    API    Resources    Community    Why TensorFlow    Search    Language    GitHub    Sign in

## Text

Overview    Tutorials    Guide    API

Filter

### TEXT GENERATION

Generate Text with RNNs    **Translate text with seq2seq models**    Translate text with transformer models

### TEXT CLASSIFICATION

Text classification with BERT    Text classification with RNNs    Compute Similarity Metrics

### NLP WITH BERT

Fine tune BERT    Fine tune BERT with GLUE    Quantify uncertainty with BERT

## Neural machine translation with attention

Was this helpful?

Run in Google Colab    View source on GitHub    Download notebook

This notebook trains a sequence to sequence (seq2seq) model for Spanish to English translation based on [Effective Approaches to Attention-based Neural Machine Translation](#). This is an advanced example that assumes some knowledge of:

- Sequence to sequence models
- TensorFlow fundamentals below the keras layer:
  - Working with tensors directly
  - Writing custom `keras.Model`s and `keras.layers`

While this architecture is somewhat outdated it is still a very useful project to work through to get a deeper understanding of attention mechanisms (before going on to [Transformers](#)).

After training the model in this notebook, you will be able to input a Spanish sentence, such as "*¿todavía estan en casa?*", and return the English translation: "are you still at home?"

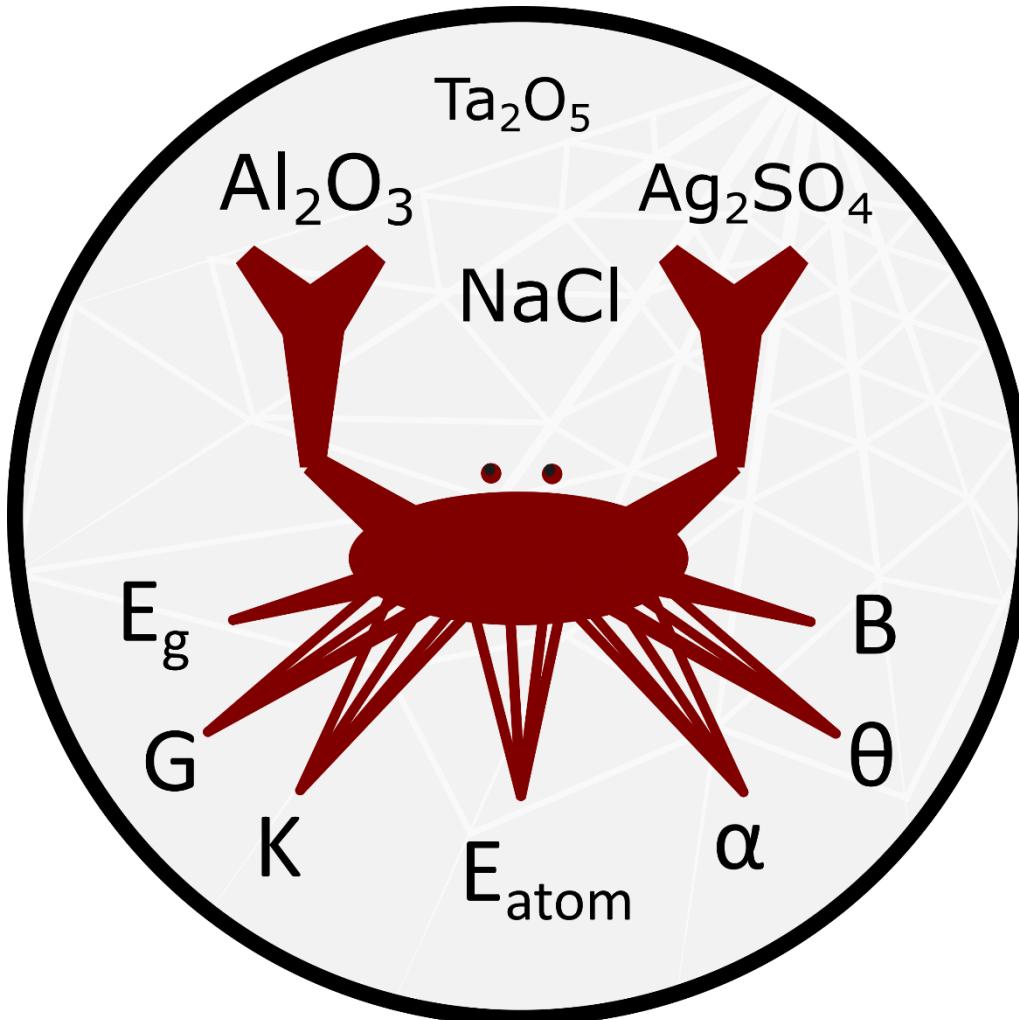
The resulting model is exportable as a `tf.saved_model`, so it can be used in other TensorFlow environments.

The translation quality is reasonable for a toy example, but the generated attention plot is perhaps more interesting. This shows which parts of the input sentence has the model's attention while translating:

On this page

- Setup
- Shape checker
- The data
  - Download and prepare the dataset
  - Create a tf.data dataset
  - Text preprocessing
- The encoder/decoder model
  - The encoder
  - The attention head
  - Test the Attention layer
  - The decoder
- Training
  - Define the loss function
  - Implement the training step
  - Test the training step
  - Train the model
- Translate
  - Convert token IDs to text
  - Sample from the decoder's predictions
  - Implement the translation loop
  - [Optional] Use a symbolic loop
  - Visualize the process
  - Labeled attention plots
- Export
- Next steps

# Compositionally-restricted attention-based network (CrabNet)



ARTICLE OPEN

## Compositionally restricted attention-based network for materials property predictions

Anthony Yu-Tung Wang  <sup>1,3</sup>, Steven K. Kauwe <sup>2,3</sup>, Ryan J. Murdock <sup>2</sup> and Taylor D. Sparks  <sup>2</sup> 

In this paper, we demonstrate an application of the Transformer self-attention mechanism in the context of materials science. Our network, the Compositionally Restricted Attention-Based network (CrabNet), explores the area of structure-agnostic materials property predictions when only a chemical formula is provided. Our results show that CrabNet's performance matches or exceeds current best-practice methods on nearly all of 28 total benchmark datasets. We also demonstrate how CrabNet's architecture lends itself towards model interpretability by showing different visualization approaches that are made possible by its design. We feel confident that CrabNet and its attention-based framework will be of keen interest to future materials informatics researchers.

*npj Computational Materials* (2021) 7:77; <https://doi.org/10.1038/s41524-021-00545-1>

### INTRODUCTION

Materials scientists constantly strive to achieve better understanding, and therefore better predictions, of materials properties. This began with the collection of empirical evidence through repeated experimentation, resulting in mathematical generalizations, theories, and laws. More recently, computational methods have arisen to solve a large variety of problems that were intractable to analytical approaches alone<sup>1,2</sup>.

As experimental and computational methods have become more efficient, high-quality data has opened up a new avenue to materials understanding. Materials informatics (MI) is the resulting field of research that utilizes statistical and machine learning (ML) approaches in combination with high-throughput computation to analyze the wealth of existing materials information and gain unique insights<sup>3–4</sup>. As this wealth has increased, practitioners of MI have increasingly turned to deep learning techniques to model and represent inorganic chemistry, resulting in approaches such as EleNet, IRNet, CGCNN, SchNet, and Roost<sup>5–9</sup>. In specific cases including CGCNN and SchNet, the compounds are represented using their chemical and structural information<sup>7,10–15</sup>.

Modeling approaches based on crystal structure are an excellent tool for MI. Unfortunately, there are many material property datasets that lack suitable structural information. An example of this is the experimental band gap data gathered by Zhou et al.<sup>16</sup>. Conversely, many databases such as the Inorganic Crystal Structure Database (ICSD) and Pearson's Crystal Data (PCD) contain an abundance of structural information, but lack the associated material properties of the recorded structures. In both cases, the applicability of structure-based learning approaches are limited. This limitation is particularly evident in the discovery of novel materials, since it is not possible to know the structural information of (currently undiscovered) chemical compounds *a priori*. Therefore, the development of structure-agnostic techniques is well-suited to the discovery of novel materials.

A typical approach to structure-agnostic learning is done by representing chemistry as a composition-based feature vector (CBFV)<sup>17</sup>. This allows for data-driven learning in the absence of structural information. The CBFV is a common way to transform chemical compositions into usable features for ML and is

generated from the descriptive statistics of a compound's constituent element properties. Researchers have effectively used CBFV-based ML techniques to generate material property predictions<sup>7–25</sup>.

One potential issue with the CBFV approach lies in the way the element vectors are combined to form the vector describing the chemical compound. Typically, the individual element vectors of the compound are scaled by the element's prevalence (fractional abundance) in the composition, before being used to form the CBFV. This step assumes that the stoichiometric prevalence of constituent elements in a compound dictates their chemical signal, or contribution, to the material's property. However, this is not true in all cases; an extreme example of this is element doping. Dopants can be present in very small amounts in a compound, but can have a significant impact on its electrical<sup>23,26,27</sup>, mechanical<sup>20,28–30</sup> and thermal properties<sup>31–34</sup>. In the case of a typical CBFV approach that uses the weighted average of element properties as a feature, the signal from dopant elements would not significantly change the vector representation of a compound. As a result, the trained ML model would fail to capture a portion of the relevant chemical information available in the full composition.

It is apparent that there is no generally accepted best way to model materials property behaviors. Different ML approaches lend themselves towards different modeling tasks. CGCNN requires access to structural information, EleNet operates within the realm of large data, and classical models excel when domain knowledge can be exploited to overcome data scarcity<sup>35</sup>. To address the diversity of learning challenges, in Dunn et al., the Automatminer framework uses computationally expensive searches to optimize classical modeling techniques. They demonstrate effective learning on some data, but show shortcomings when deep learning is appropriate<sup>36</sup>.

In a similar spirit, we seek to overcome general challenges in the area of structure-agnostic learning using an approach we refer to as the Compositionally Restricted Attention-Based network (CrabNet). CrabNet introduces the self-attention mechanism to the task of materials property predictions, and dynamically learns and updates individual element representations based on their chemical environment. To enable this, we introduce a

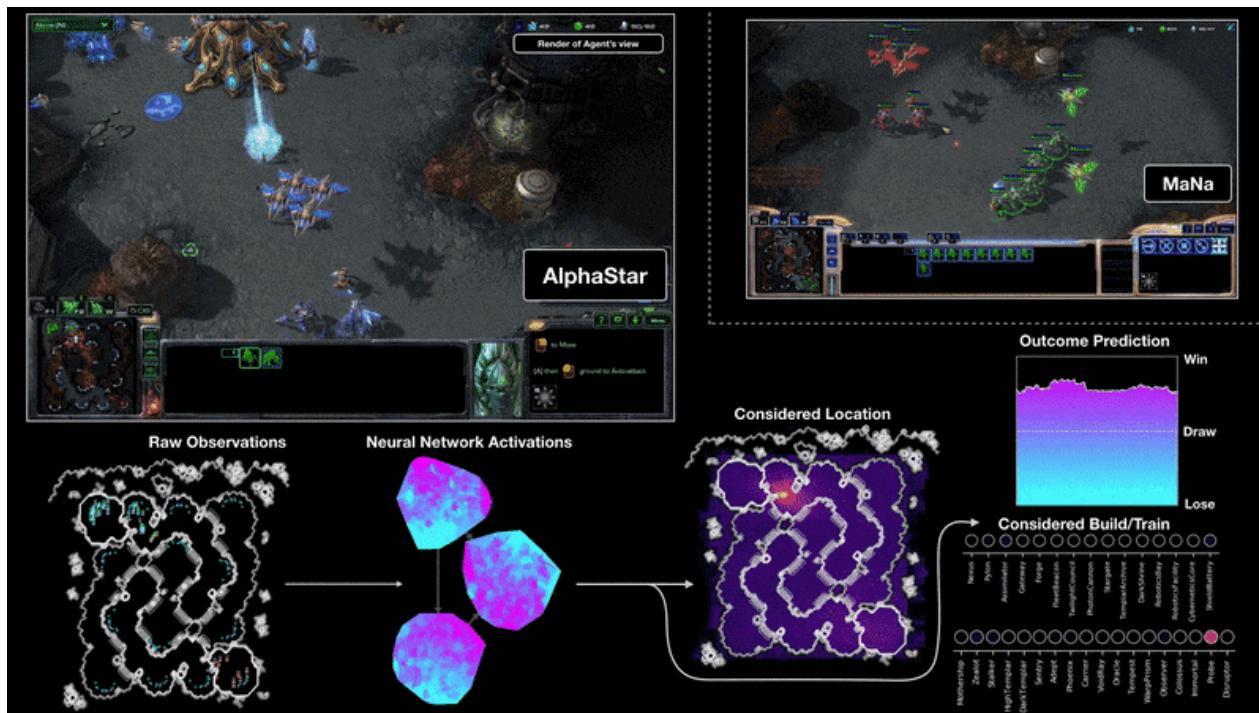
<sup>1</sup>Fachgebiet Keramische Werkstoffe/Chair of Advanced Ceramic Materials, Technische Universität Berlin, Berlin, Germany. <sup>2</sup>Department of Materials Science & Engineering, University of Utah, Salt Lake City, UT, USA. <sup>3</sup>These authors contributed equally: Anthony Yu-Tung Wang, Steven K. Kauwe. Email: sparks@eng.utah.edu

CrabNets are transformers in disguise!

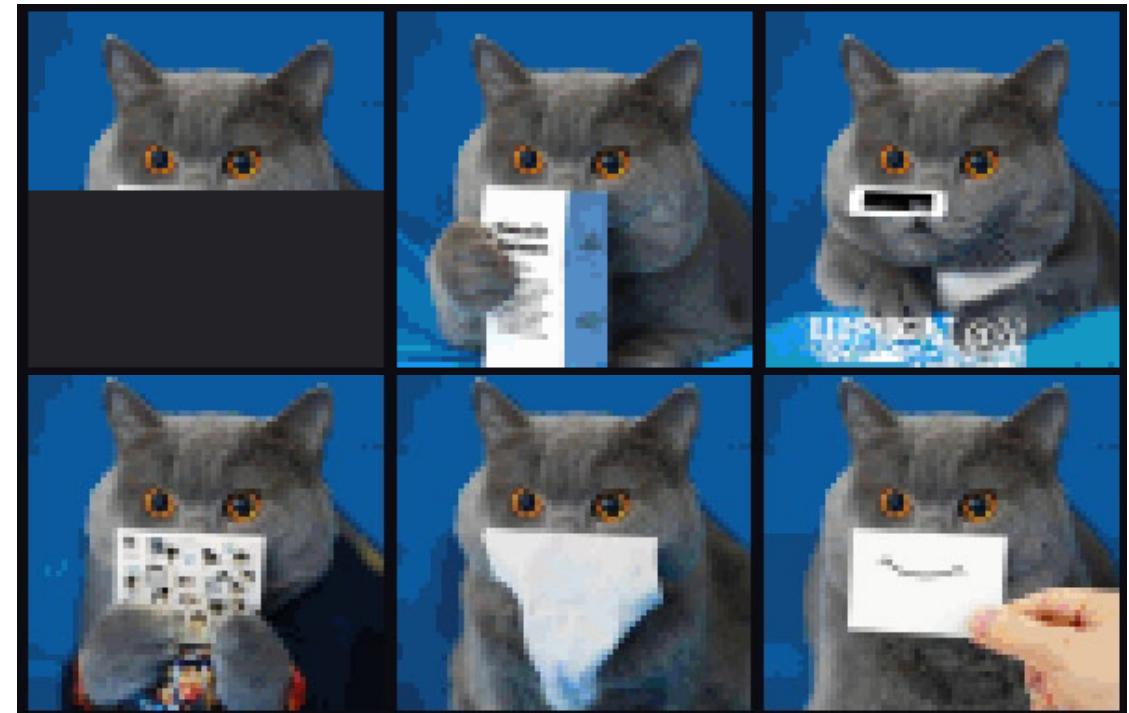


The transformer is behind the NLP revolution

## Deepmind's AlphaStar AI



## Image GPT (complete the image)



What are the benefits of attention

Well optimized learning

Per-element representations

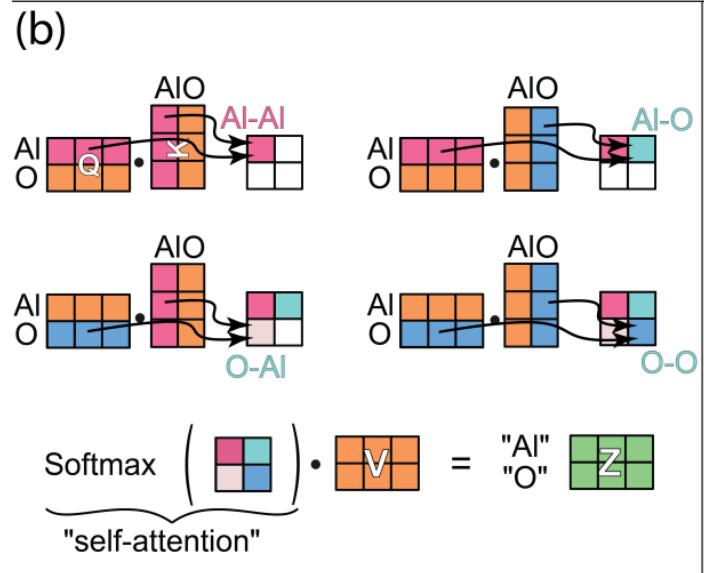
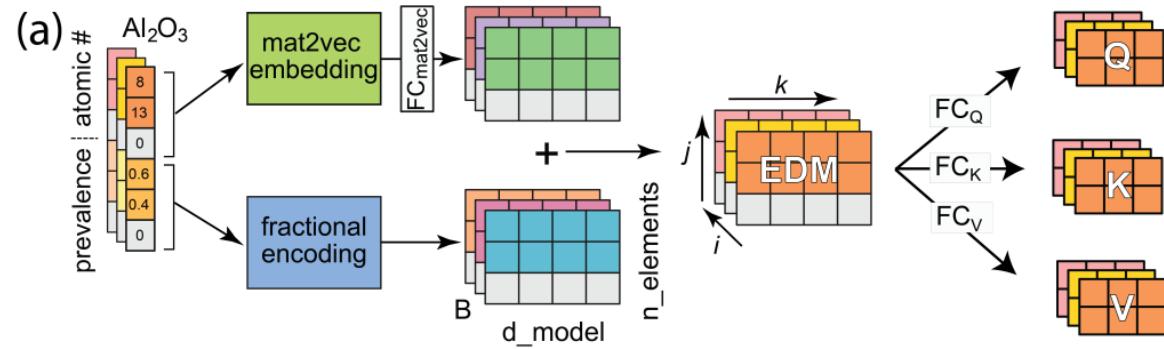
Element-element interactions

Attention maps!

Element property contributions

Permutation invariant

# The attention blocks are incorporated into a neural network architecture

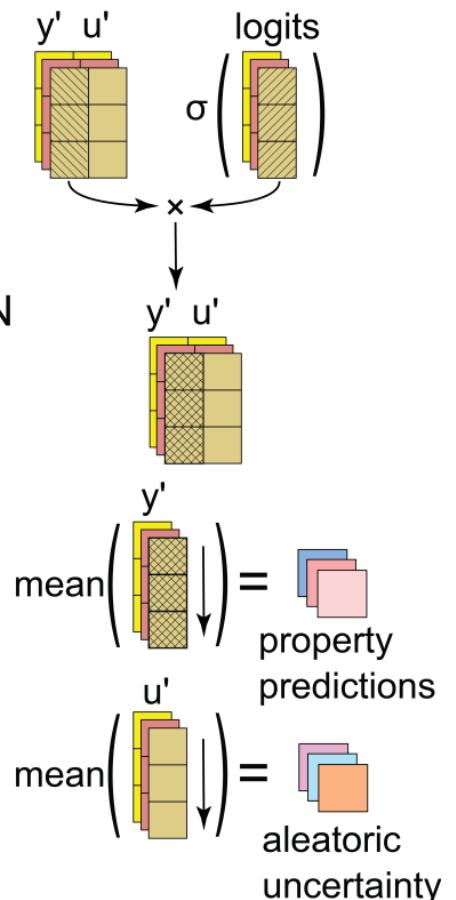
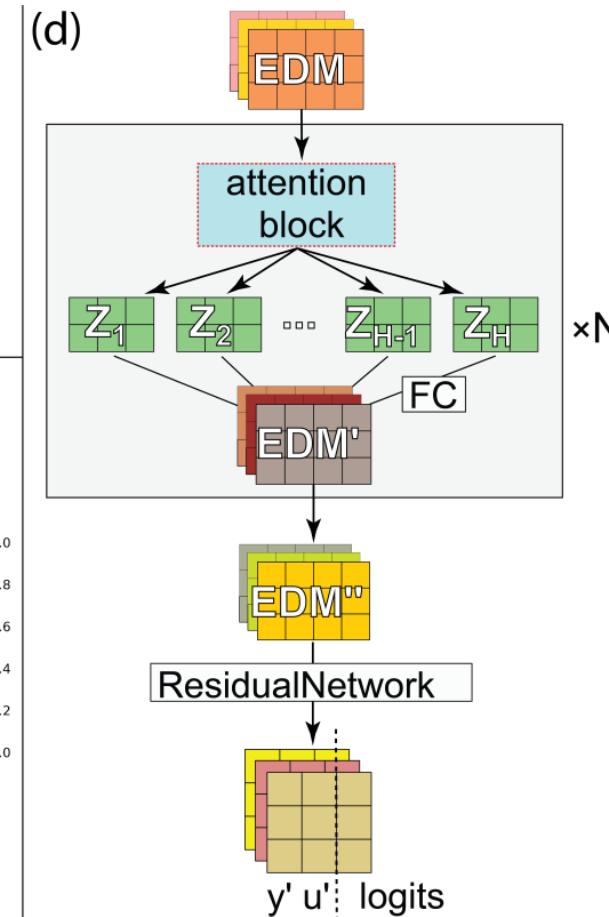


(c) Attention weight matrices. Four 4x4 attention weight matrices are shown, corresponding to the four element pairs: O-O, Al-Al, None-None, and O-Al. The matrices are color-coded based on their values, with a color scale from 0.0 (light yellow) to 1.0 (dark purple).

	O	Al	None	None
O	0.71	0.29	0	0
Al	0.92	0.08	0	0
None	0.01	0.99	0	0
None	0.82	0.18	0	0
O	0	0	0	0
Al	0	0	0	0
None	0	0	0	0
None	0	0	0	0

	O	Al	None	None
O	0.57	0.43	0	0
Al	0.99	0.01	0	0
None	0.08	0.92	0	0
None	0.29	0.71	0	0
O	0	0	0	0
Al	0	0	0	0
None	0	0	0	0
None	0	0	0	0



# CrabNet reaches state-of-the-art performance

## Performance comparison

MatBench Properties	Roost	CrabNet	HotCrab	ElemNet	RF
Castelli perovskites	0.359	0.408	0.410	0.468	0.581
Refractive index	0.327	0.309	0.323	0.538	0.419
Shear modulus (log10)	0.104	0.097	0.102	0.140	0.105
Bulk modulus (log10)	0.078	0.073	0.077	0.124	0.082
Experimental band gap	0.374	0.339	0.353	0.450	0.444
DFT Exfoliation energy	47.692	45.617	48.943	57.673	50.000
MP Formation energy	0.082	0.083	0.085	1.029	0.121
MP Band gap	0.252	0.258	0.265	0.337	0.328
Phonon peak	46.158	49.868	56.258	nan	64.924
Steels yield	155.231	91.783	92.507	nan	104.538

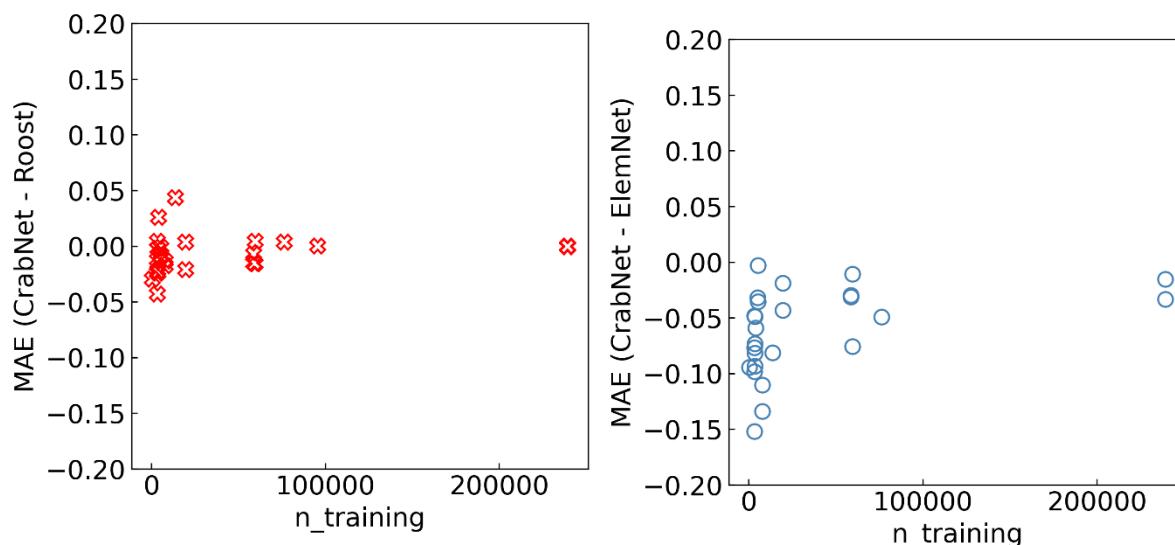
Extended Properties	Roost	CrabNet	HotCrab	ElemNet	RF
AFLOW Bulk modulus	8.988	8.861	9.271	12.242	12.006
AFLOW Debye temperature	37.478	33.717	36.027	47.937	36.352
AFLOW Shear modulus	10.063	9.101	9.510	12.890	10.125
AFLOW Thermal conductivity	2.697	2.316	2.251	3.383	2.724
AFLOW Thermal expansion	3.68e-06	3.84e-06	3.87e-06	6.36e-06	5.54e-06
AFLOW Band gap	0.337	0.302	0.316	0.375	0.384
AFLOW Energy per atom	0.086	0.093	0.094	0.129	0.230
Bartel Decomposition (Ed)	0.067	0.063	0.066	0.081	0.090
Bartel Formation (Ef)	0.055	0.059	0.059	0.071	0.100
MP Bulk modulus	14.950	14.264	14.982	17.236	16.754
MP Elastic anisotropy	9.273	9.233	9.533	9.500	11.239
MP Energy above convex hull	0.089	0.086	0.089	0.100	0.113
MP Magnetic moment	2.583	2.395	2.433	2.761	2.664
MP Shear modulus	12.674	12.363	13.068	15.374	13.368
OQMD Band gap	nan	0.039	0.038	0.119	0.049
OQMD Energy per atom	0.035	0.035	0.035	0.065	0.145
OQMD Formation enthalpy	0.033	0.033	0.032	0.062	0.085
OQMD Volume per atom	0.265	0.247	0.247	nan	0.529

# CrabNet reaches state-of-the-art performance

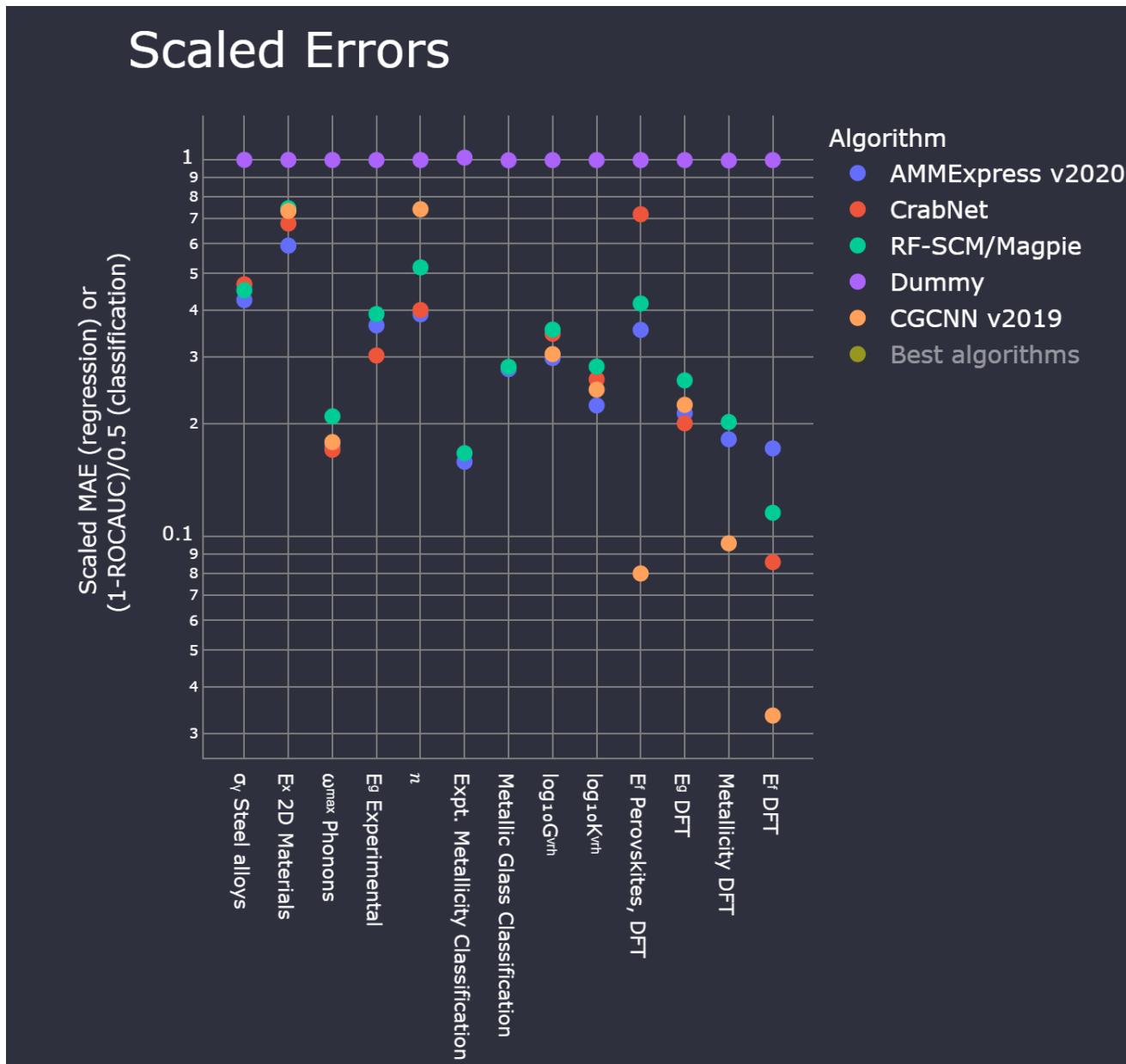
## Performance comparison

MatBench Properties	Roost	CrabNet	HotCrab	ElemNet	RF
Castelli perovskites	0.359	0.408	0.410	0.468	0.581
Refractive index	0.327	0.309	0.323	0.538	0.419
Shear modulus (log10)	0.104	0.097	0.102	0.140	0.105
Bulk modulus (log10)	0.078	0.073	0.077	0.124	0.082
Experimental band gap	0.374	0.339	0.353	0.450	0.444
DFT Exfoliation energy	47.692	45.617	48.943	57.673	50.000
MP Formation energy	0.082	0.083	0.085	1.029	0.121
MP Band gap	0.252	0.258	0.265	0.337	0.328
Phonon peak	46.158	49.868	56.258	nan	64.924
Steels yield	155.231	91.783	92.507	nan	104.538

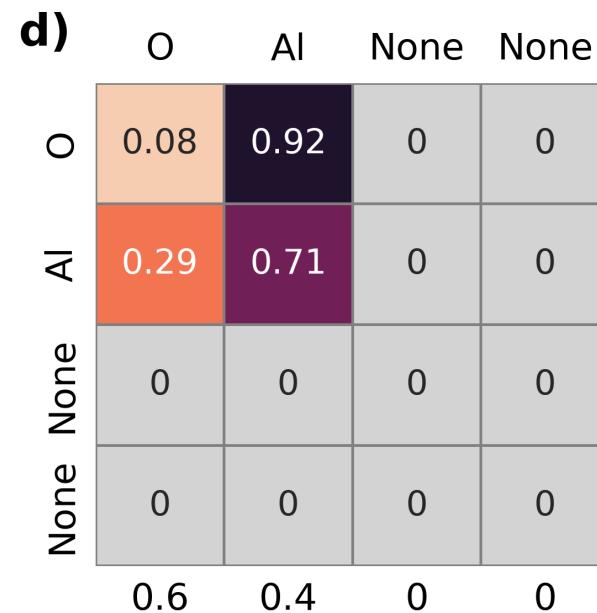
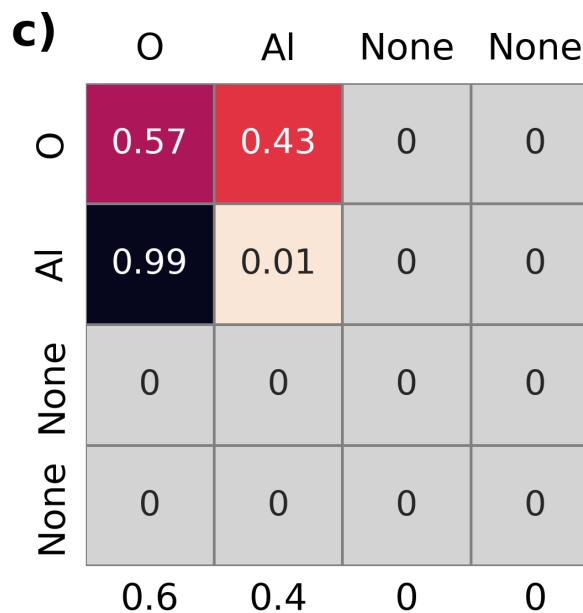
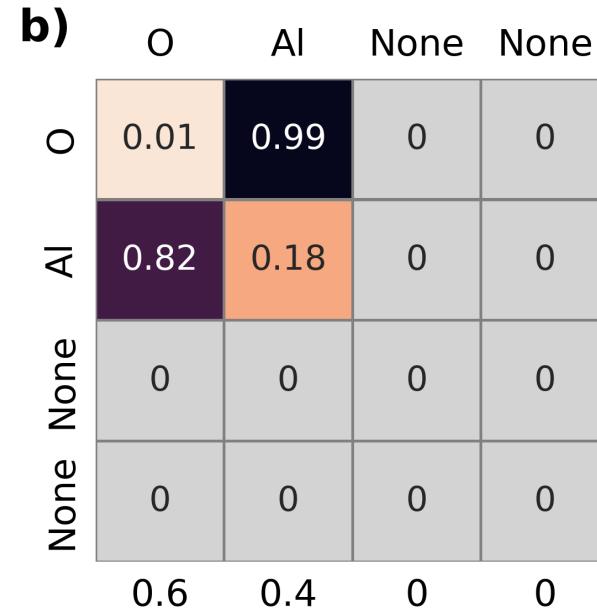
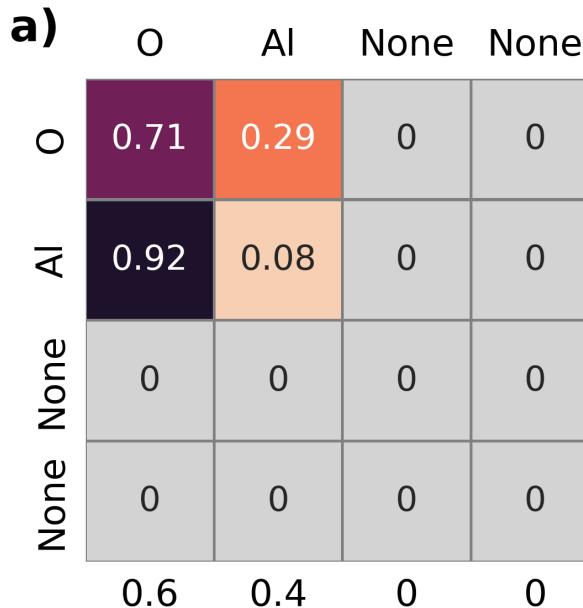
Extended Properties	Roost	CrabNet	HotCrab	ElemNet	RF
AFLOW Bulk modulus	8.988	8.861	9.271	12.242	12.006
AFLOW Debye temperature	37.478	33.717	36.027	47.937	36.352
AFLOW Shear modulus	10.063	9.101	9.510	12.890	10.125
AFLOW Thermal conductivity	2.697	2.316	2.251	3.383	2.724
AFLOW Thermal expansion	3.68e-06	3.84e-06	3.87e-06	6.36e-06	5.54e-06
AFLOW Band gap	0.337	0.302	0.316	0.375	0.384
AFLOW Energy per atom	0.086	0.093	0.094	0.129	0.230
Bartel Decomposition (Ed)	0.067	0.063	0.066	0.081	0.090
Bartel Formation (Ef)	0.055	0.059	0.059	0.071	0.100
MP Bulk modulus	14.950	14.264	14.982	17.236	16.754
MP Elastic anisotropy	9.273	9.233	9.533	9.500	11.239
MP Energy above convex hull	0.089	0.086	0.089	0.100	0.113
MP Magnetic moment	2.583	2.395	2.433	2.761	2.664
MP Shear modulus	12.674	12.363	13.068	15.374	13.368
OQMD Band gap	nan	0.039	0.038	0.119	0.049
OQMD Energy per atom	0.035	0.035	0.035	0.065	0.145
OQMD Formation enthalpy	0.033	0.033	0.032	0.062	0.085
OQMD Volume per atom	0.265	0.247	0.247	nan	0.529



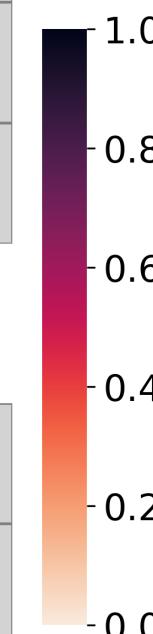
CrabNet on par with algorithms that include both structure and composition!



Attention maps allow us to visualize where the algorithms attends to for predictions



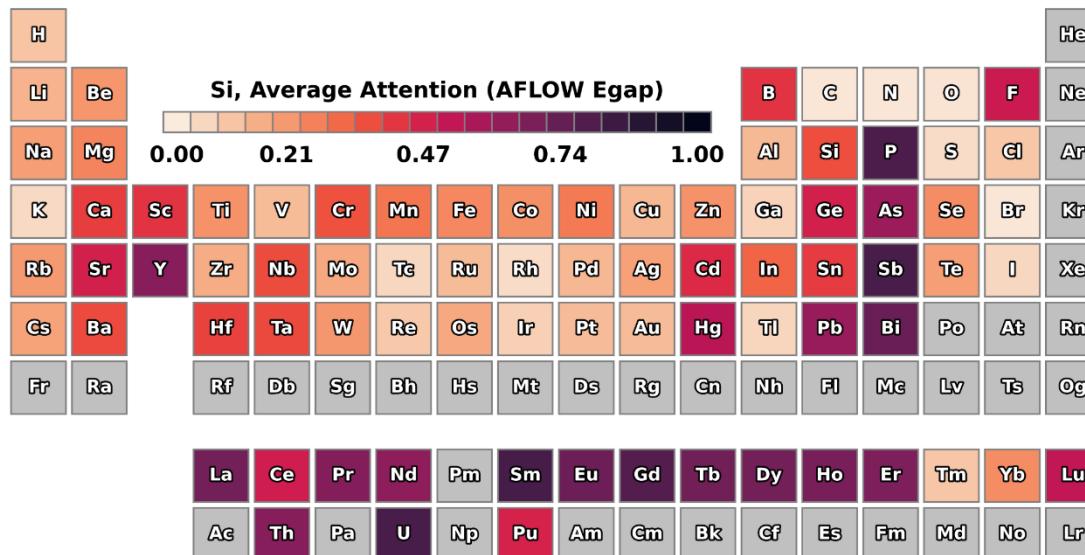
# $\text{Al}_2\text{O}_3$ Bulk Modulus



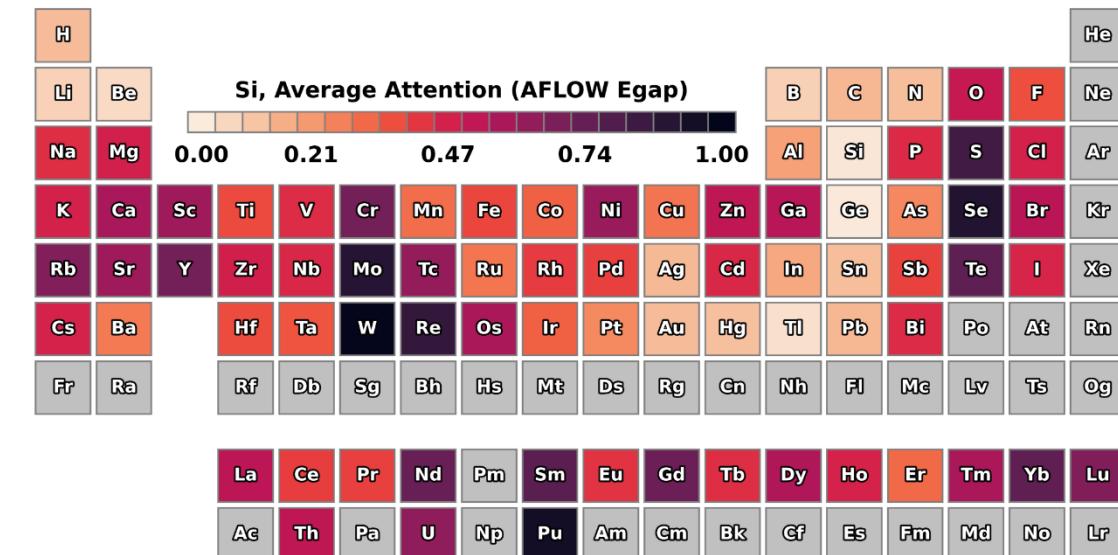
Attention heads *after training.*

Attention maps allow us to visualize where the algorithms attends to for predictions

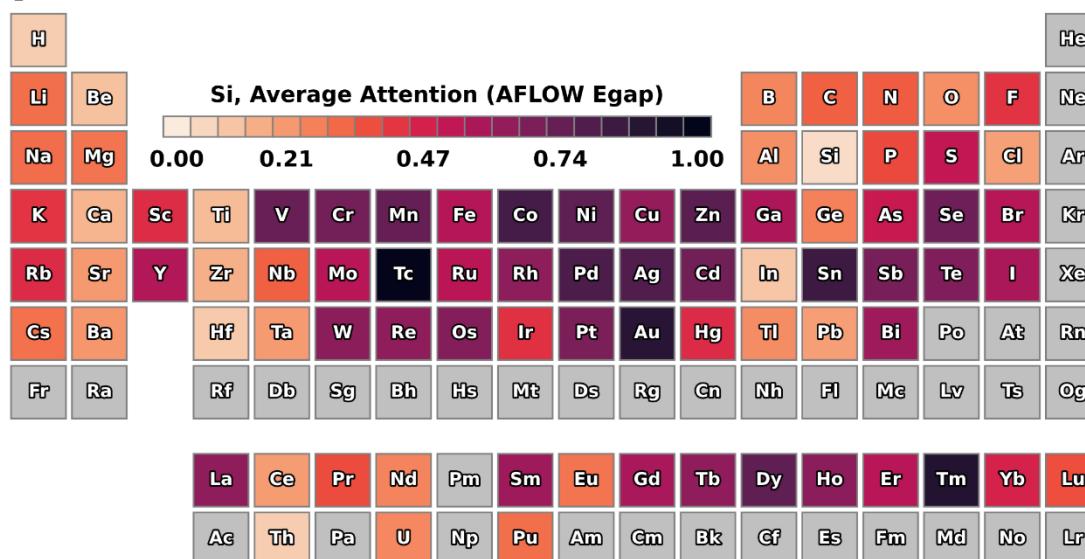
a)



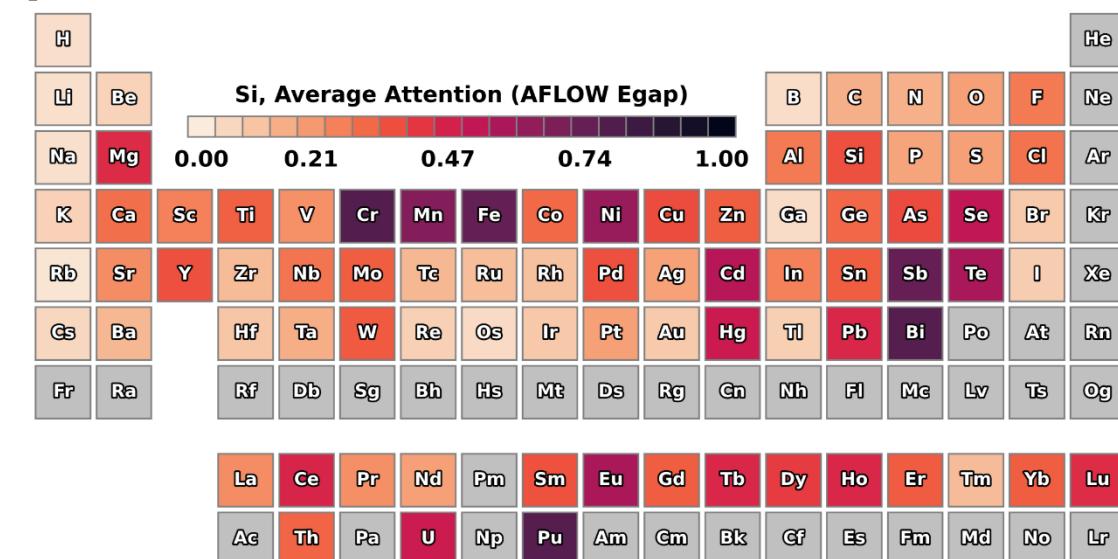
b)



c)



d)



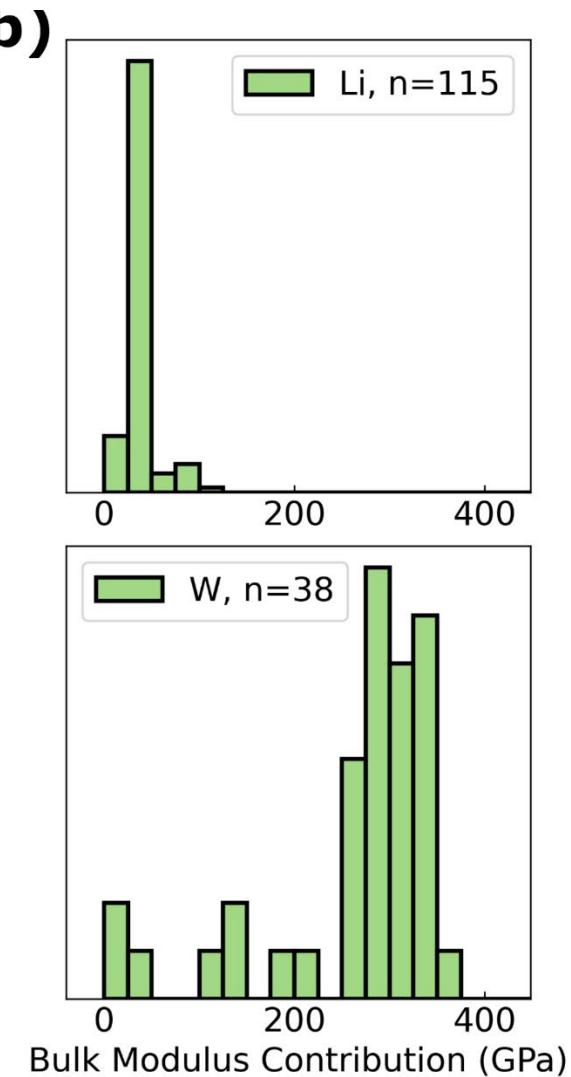
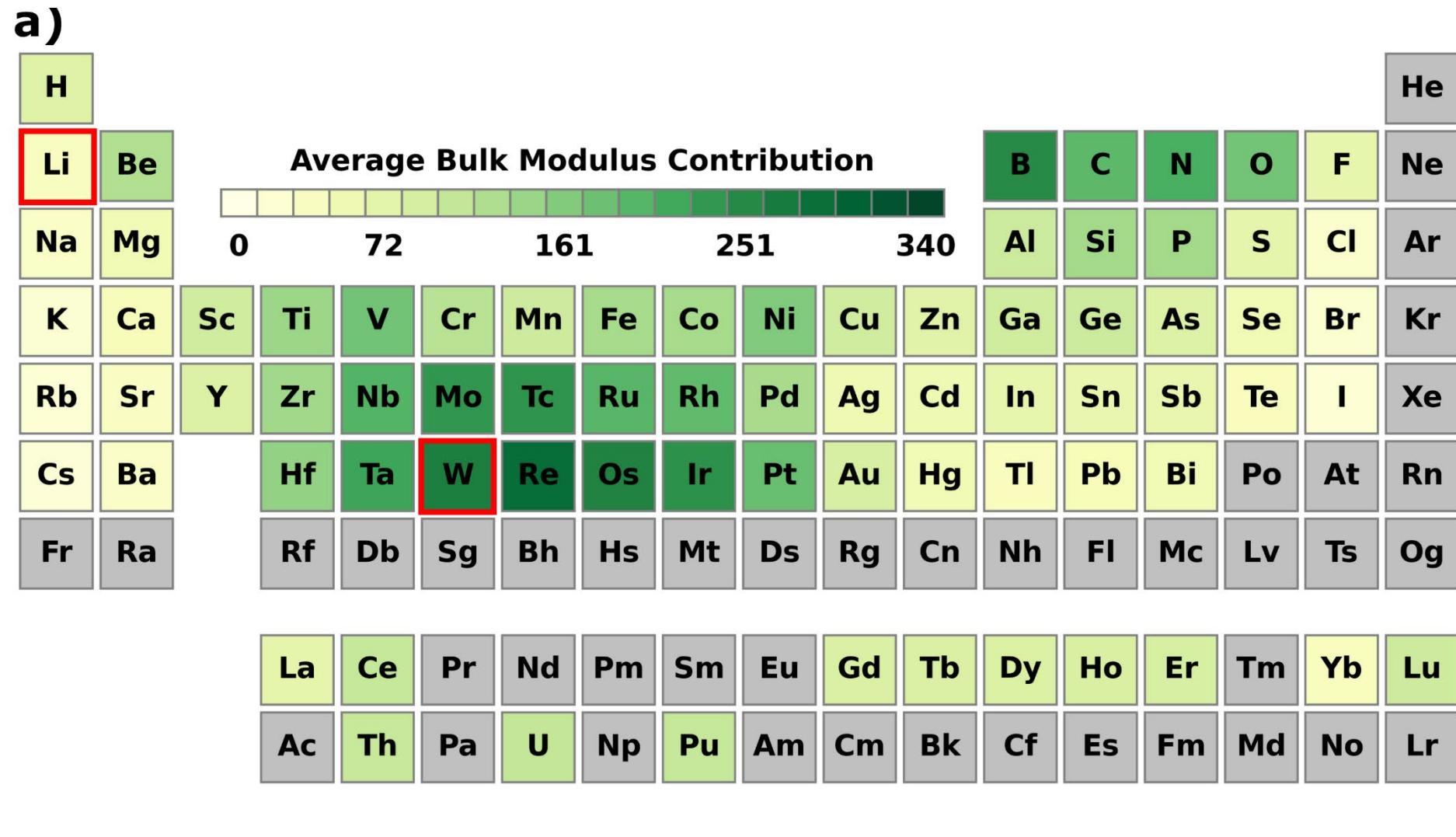
Maps are generated during training, so we can create videos of how they change



Maps are generated during training, so we can create videos of how they change

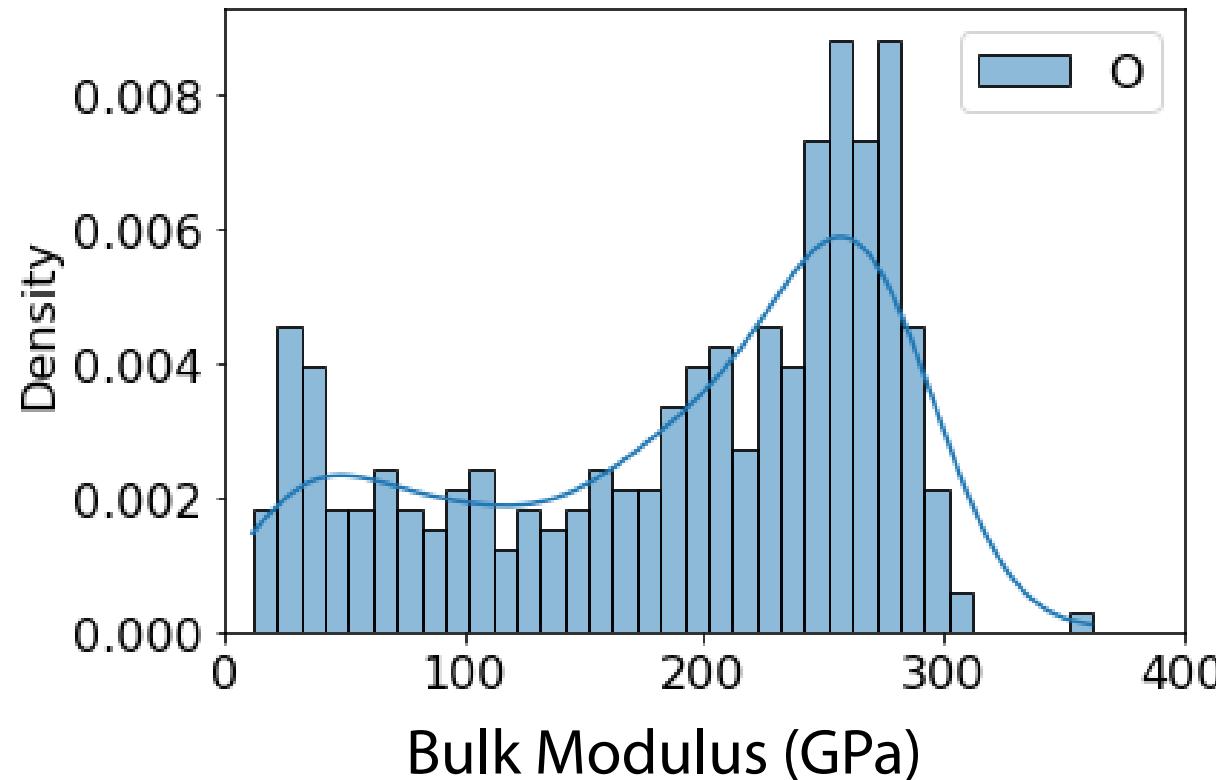


Visualization of element contributions offers better understanding and better expectations

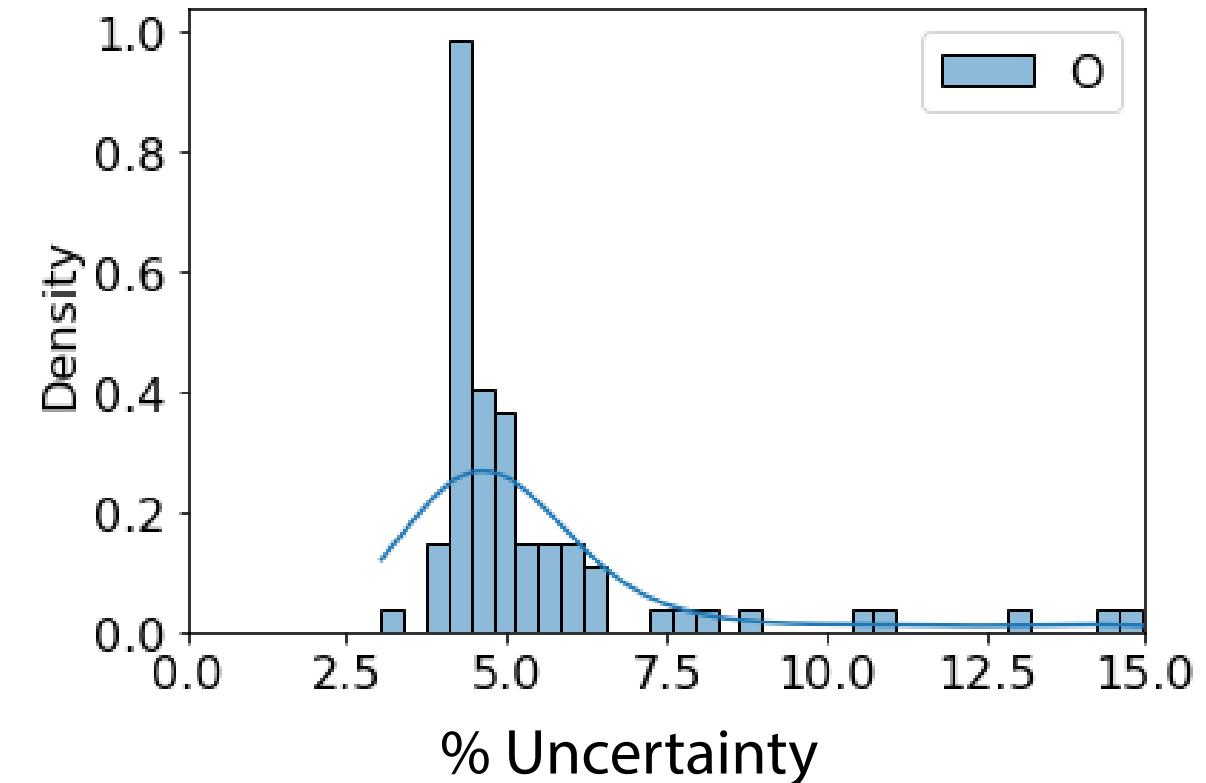


We can also see elemental contribution to a property and uncertainty

**Element bulk modulus contribution**



**Element uncertainty contribution**



# generative machine learning models

