Structure recognition with graph neural networks

An intermediate report for the course "Advanced Projects in Computational Physics 2"

From
Stephen Weybrecht
December 9, 2024

Supervisor: Jonas Buba

Abstract

The project described in the following lies at the intersection of solid state physics and machine learning. On the one hand there is the physical problem, namely the classification of noisy crystal lattices in 2 and 3 dimensions into their corresponding Bravais lattice group. For this graphs are randomly generated in a first step. These graphs then need to be classified efficiently and robustly, regardless of noise and introduced defects. As is the case with other classification tasks, neural networks promise an interesting approach to this goal and will therefore be the second part of this project. For this special networks designed for handling graph like structures, so called Graph convolutional networks are employed. These use a concept called message passing to efficiently enable graph level classification tasks.

By use of these concepts a test accuracy of over 90% has been achieved for both the 2D and 3D case. Future goals include testing out other features and network architectures for this classification task, as well as the specialisation in defect detection in mono atomic crystal lattices.

Contents

1 Theoretical introduction

1.1 Bravais lattices

In the first part of the project, our task was to deal with crystal lattices in 2 and 3 dimensions. For the following discussion an introduction of the terms "crystal", "basis" and "Bravais lattice" is therefore needed.

Following the discussion of [1] an ideal crystal is a periodic, infinite arrangement of atoms in a solid. These atoms are arranged in blocks, a so called basis, in a regularly spaced grid, the lattice. In other words the lattice represents a schema after which individual atoms or groups of atoms (the basis) are arranged to form the crystal. A lattice in d dimensions can be defined by a set of d translation vectors. The superposition of integer multiples of these vectors then makes up the lattice [1]. In principle the length and direction of these vectors can be arbitrary. In this case the lattice would generally not map into itself under translations and rotations – the lattice is called oblique. There are however special sets of translation vectors which form lattices of high symmetry. These fundamental lattices are called the Bravais lattices. For $d = 2$ there are 5 (4 special and one oblique) Bravais lattices, while for $d = 3$ there are 14 (13 special cases and one oblique, so called triclinic lattice). These are depicted in Figure 1 and Figure 2 respectively.

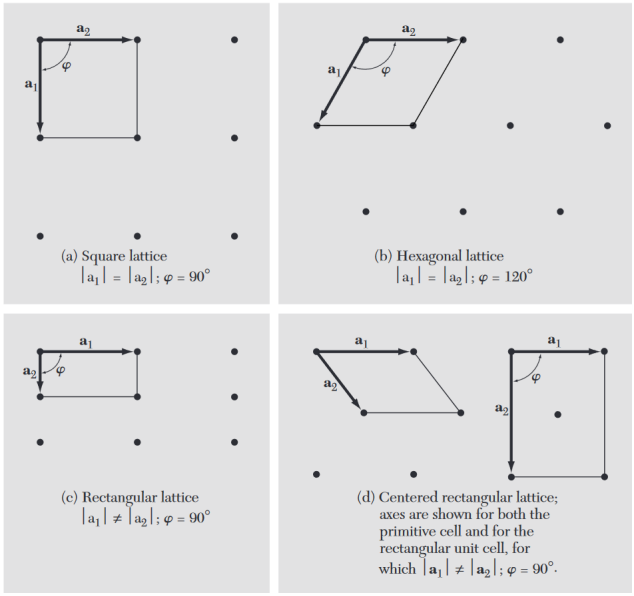

Figure 1: The five Bravais lattices for 2 dimensions. The graphic also illustrates the length of the translation vectors $\boldsymbol{a}_i$ and the angle between them in order to make up the corresponding lattice. Taken from [1, Fig. 7].

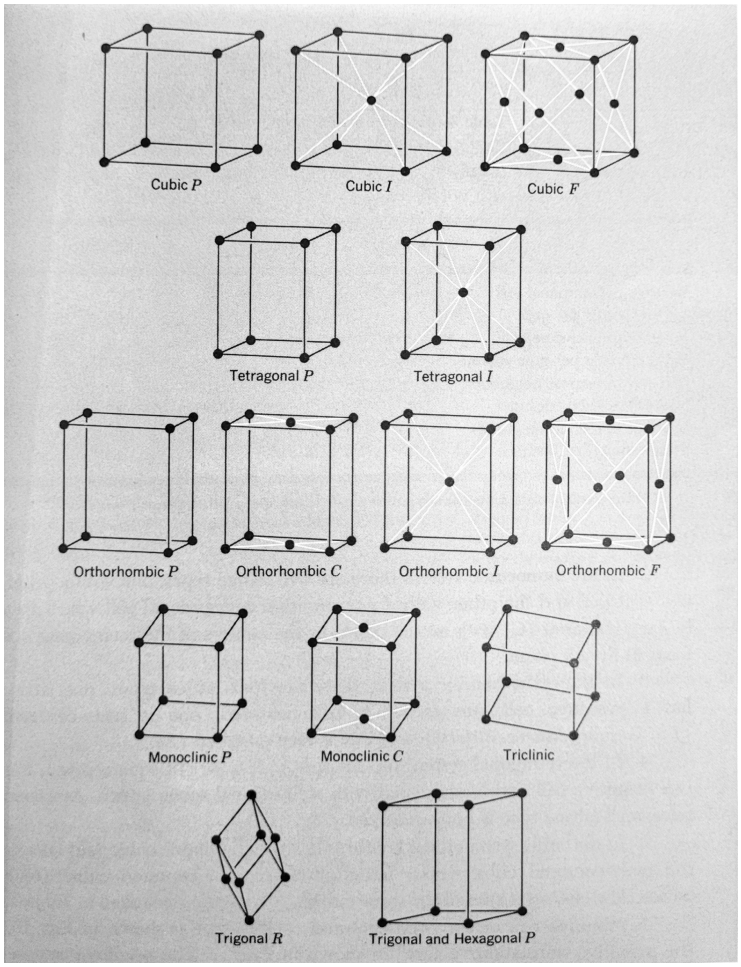

Figure 2: The 14 Bravais lattices for the $d = 3$ case. One further classifies these lattices into seven types of cells: Cubic (a=b=c, α=β=γ=90°), Tetragonal (a=b≠c, α=β=γ=90°), Orthorhombic (α=β=γ=90°), Monoclinic (α=β=90°≠γ), Triclinic, Triagonal (a=b=c, α=β=γ < 120°≠90°), Hexagonal (a=b≠c, α=β=90°, γ=120°). Note that in the prior notation a, b and c denote the lengths of the translation vectors, α, β, γ the angles between them and omitted length or angle relations mean they can be arbitrary. These groups are again subclassified based on their lattice structure into simple "P", body-centred "I", base-centred "C", face-centred "F". As mentioned, all lattices are special cases of the general, triclinic case. The graphic is taken from [2, Fig. 14].

1.2 Graph neural networks

The lattices as discussed in the prior section are a collection of atoms linked by bonds and can therefore be suitably represented by a graph, consisting of nodes and edges. If we want to apply machine learning to crystal lattices, we therefore need models that are well suited for data organized in a graph-like manner. What follows is a basic introduction to such networks, specifically graph convolutional neural networks (GCNs).

GCNs take inspiration in the already well-established convolutional neural networks in which a typical layer consists of a trainable kernel that can be applied on ordered, grid-like training data of arbitrary size and shape (e.g.. images) [3]. GCNs represent a generalisation of this concept onto unordered nodes with a variable number of neighbours. Each GCN-layer uses so called message passing in order to update the node state $h_u^{(t)}$ of a time t to the next step $h_u^{(t+1)}$ as shown in Equation 1 [3, eq. 4.1].

$$h_u^{(t+1)} = UPDATE^{(t)}\left(h_u^{(t)}, AGGREGATE^{(t)}\left(\left\{h_v^{(t)}, \forall v \in N(u)\right\}\right)\right) \tag{1}$$

In the above equation $UPDATE^{(t)}$ and $AGGREGATE^{(t)}$ could be any differentiable functions i.e. also neural networks, and $N(u)$ denotes the neighbourhood of u meaning all directly connected nodes in the graph. This equation implies the following update schema that is also depicted in Figure 3:
The starting point is a graph, consisting of nodes with feature vectors and connections, that could also have features. For each node, messages from neighbouring nodes are aggregated to a single message by taking a weighted mean of the neighbouring nodes feature vectors. This operation can also be weighted by use of edge features. This message is then passed to a non-linear update function (e.g.. ReLU), that updates the node in question for the next time step. After the update is completed, further operations can be performed depending on the wanted classification scheme. In the following graph classification is used, which means all feature vectors are pooled in a last step, in order to get a single quantity that is descriptive of the entire graph [3].

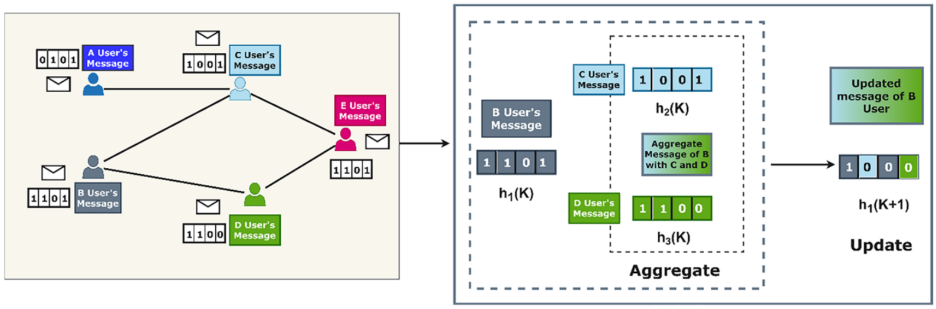

Figure 3: A graphic representation of the update schema for message passing graph neural networks. Taken from [3, Fig. 7].

2 Results so far

The main tasks so far were to gain familiarity with a machine learning Python library by working on graph classification of 2- and 3-dimensional Bravais lattices. For this I chose PyTorch, as it has a library for working with graph neural networks called PyTorch Geometric (PyG) built on top of it. The first step for both cases was to generate the training data. For this ideal Bravais lattices were created in a first step, to which random (Gaussian) noise, and defects (meaning a removal of a random number of nodes at random positions) were added. For further variety in training samples, graphs were additionally scaled by random constants to provide non-uniform node-spacing within one Bravais lattice group. After the nodes have been placed and noise was added, the connections between nodes are determined by searching for the neighbours within a given radius (that also depends on the noise amplitude) and connecting all nodes, that lie within said radius. By these means the random noise and defects are also included within the structure of the graph and have an influence in message passing.
The size of the graphs have been chosen as follows. For the plane lattices a size of 10 by 10 nodes has been chosen in order to give the network enough information about each training sample to learn something about the Bravais lattice. For the 3D lattices a size of 10 by 10 by 10 nodes has been tried but deemed unpractical as the time effort for training is quite large. This in turn leads to only being able to use a low number of training samples which leads to low accuracies of around 60%. A better approach proved to be using graphs of size 5 by 5 by 5 nodes which cuts the total number of nodes per graph by a factor of 8 while still giving the network enough information to determine the lattice correctly.
For node features the number of neighbouring nodes has been chosen while each edge has the (two or three dimensional) connection vector between its corresponding nodes as its feature. This has proven itself to be enough information to classify a test dataset of random graphs with more than 90% accuracy for both cases. For classification each lattice was labeled by its one-hot encoded type that was then compared by using the cross entropy loss function during training. As connection vectors carry quite a lot of information, for future tests other features like connection lengths or binding angles are planned in order to see, how much information about the lattice is needed to suitably classify it. For these preliminary results a high degree of information in the features and shallow neural network (2D: 2 GINEConv layers [4] with a total of 12 hidden neurons, 3D 3 GINEConv layers with 50 hidden neurons total) was however used as a proof of concept and to see whether graph generation worked.

3 Plans for the future

In the following the goals for the remaining part of the project will be discussed and a rough time estimate for each step will be given. As graph generation and classification has worked rather well so far, I expect only little changes to be made to the already existing code. Further things to improve or try out with the existing method are varying node and edge features and choosing different layers or a different network structure. By this investigation one could find out the minimal

amount of information the network needs to classify Bravais lattices and the most efficient network architecture. As the tasks discussed so far however only served as a kind of general introduction into the actual topic that is specific to me, extensive experiments with the old goal of lattice classification are not planned. Instead the focus shifts towards a new goal, namely the detection of defects in mono atomic crystals. I expect to be able to reuse significant parts of the lattice generation code I have written so far, although significant changes in the network architecture and features will most likely be made. A meeting with Jonas Buba for discussing the specifics for the future project is scheduled for Wednesday, the 11.12. After that I plan to finish the remaining experiments having to do with the old goal of Bravais lattice classification until the end of December. During the same time I want to gain familiarity with the theory and suitable approaches for defect detection by reading literature and starting to code. Until the mid of January I plan to finalise the coding part of the project, to be able to focus on writing the report and preparing the final presentation during the remaining time.

4 Code availability

The code written for this project is made available in the following Git repository: `https://github.com/SteWey0/Computerpraktikum/`

Bibliography

[1] Charles Kittel. "Chapter 1: Crystal Structure". In: *Introduction to Solid State Physics*. 8. ed. Wiley, 2005. ISBN: 978-0-471-41526-8.

[2] Charles Kittel. "Chapter 1: Crystal Structure". In: *Introduction to Solid State Physics*. 4. ed. Wiley, 1971. ISBN: 0-471-49021-0.

[3] Bharti Khemani, Shruti Patil, Ketan Kotecha, and Sudeep Tanwar. "A Review of Graph Neural Networks: Concepts, Architectures, Techniques, Challenges, Datasets, Applications, and Future Directions". In: *Journal of Big Data* 11.1 (Jan. 16, 2024), p. 18. ISSN: 2196-1115. DOI: 10.1186/s40537-023-00876-4. URL: https://doi.org/10.1186/s40537-023-00876-4 (visited on 12/08/2024).

[4] PyG Team. *GINEConv*. GINEConv. Dec. 9, 2024. URL: https://pytorch-geometric.readthedocs.io/en/latest/generated/torch_geometric.nn.conv.GINEConv.html#torch_geometric.nn.conv.GINEConv (visited on 12/09/2024).