1 Definitions

1.1 Deep learning

TODO:

1.2 Graphs

A graph G is defined as a couple (V, E) where V represents the set of nodes and $E \subseteq \binom{V}{2}$ is the set of edges connecting these nodes.
TODO: Example of figure
We encounter the notion of graphs several times in deep learning:

- Connections between two layers of a deep learning model can be represented as a bipartite graph, coined *connectivity graph*. It encodes how the information is propagated through a layer to another. See section 1.2.1.
- A computation graph is used by deep learning frameworks to keep track of the dependencies between layers of a deep learning models, in order to compute forward and back-propagation. See section 1.2.2.
- A graph can represent the underlying structure of an object (often a vector), whose nodes represent its features. See section 1.2.3.
- Datasets can also be graph-structured, where the nodes represent the objects of the dataset. See section 1.2.4.

1.2.1 Connectivity graph

A Connectivity graph is a graphical representation of the linear part of the mathematical model implemented by a layer of neurons. Formally, given a linear part of a layer, let $\mathbf{x}$ and $\mathbf{y}$ be the input and output signals, n the size of the set of input neurons $N = \{u_1, u_2, \ldots, u_n\}$, and m the size of the set of output neurons $M = \{v_1, v_2, \ldots, v_m\}$. This layer implements the equation $y = \Theta x$ where Θ is a $n \times m$ matrix.

Definition 1.1. The *connectivity graph* $G = (V, E)$ is defined such that $V = N \cup M$ and $E = \{(u_i, v_j) \in N \times M, \Theta_{ij} \neq 0\}$.

I.e. the connectivity graph is obtained by drawing an edge between neurons for which $\Theta_{ij} \neq 0$. For instance, in the special case of a complete bipartite graph, we would obtain a dense layer. Connectivity graphs are especially useful to represent partially connected layers, for which most of the Θ_{ij} are 0. For example, in the case of layers characterized by a small local receptive field, the connectivity graph would be sparse, and output neurons would be connected to a set of input neurons that corresponds to features that are close together in the input space. Figure 1 depicts some examples.
TODO: Figure 1. It's just a placeholder right now

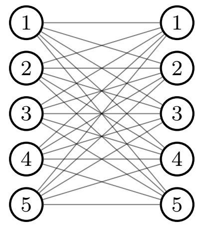

Figure 1: Examples

Connectivity graphs also allow to graphically modelize how weights are tied in a neural layer. Let's suppose the $\Theta_i j$ are taking their values only into the finite set $K = \{w_1, w_2, \ldots, w_\kappa\}$ of size κ, which we will refer to as the *kernel* of *weights*. Then we can define a labelling of the edges $s : E \to K$. s is called the *weight sharing scheme* of the layer. This layer can then be formulated as $\forall v \in M, y_v = \sum_{u \in N, (u,v) \in E} w_{s(u,v)} x_u$. Figure 2 depicts the connectivity graph of a 1-d convolution layer and its weight sharing scheme.

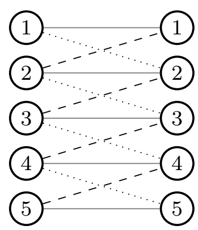

Figure 2: Depiction of a 1D-convolutional layer and its weight sharing scheme.

TODO: Add weight sharing scheme in Figure 2

1.2.2 Computation graph

1.2.3 Underlying graph structure

1.2.4 Graph-structured dataset

transductive vs inductive

1.3 Geometric grids

1.4 Grid graphs

1.5 Spatial graphs

1.6 Projections of spatial graphs