**Perceptron**

\[ f_1(w_1) + f_2(w_2) + f_3(w_3) > 0? \]

**Two-Layer Neural Network**

\[ \sum w_{12} + \sum w_{22} > 0? \]
\[ \sum w_{11} > 0? \]

**N-Layer Neural Network**

\[ \sum \sum \sum \ldots \sum h_w(f(x)) \]

**Hill Climbing**

- Simple, general idea:
  - Start wherever
  - Repeat: move to the best neighboring state.
  - If no neighbors better than current, quit.
  - Neighbors = small perturbations of \( w \).
- Property
  - Many local optima

\[ \rightarrow \text{How to find a good local optimum?} \]
Auto-Encoder (Crude Idea Sketch)

Training Procedure: Stacked Auto-Encoder

- **Auto-encoder**
  - Layer 1 = "compressed" version of input layer

- **Stacked Auto-encoder**
  - For every image, make a compressed image (= layer 1 response to image)
  - Learn Layer 2 by using compressed images as input, and as output to be predicted
  - Repeat similarly for Layer 3, 4, etc.

- **Some details left out**
  - Typically in between layers responses get agglomerated from several neurons ("pooling" / "complex cells")