4.2 ANN + Introduction to Deep Learning

Artificial Neural Networks

- Models with a strong biological inspiration.
- Composed by a set of units (neurons) that are connected. These connections have an associated weight.
- Each unit has an activation level as well as means to update this level.
- Some units are connected to the outside world. We have input and output neurons.
- Learning within ANNs consists of updating the weights of the network connections

Artificial Neuron

- Each unit has a very simple function: receive the input impulses and calculate its ouput as a function of these impulses.
- This calculation is divided in two parts:
  - a linear combination of the inputs
  - a (typically) non-linear activation function

Perceptron

- network with an input layer and an output layer
- It learns by updating the weights through delta rule with learning rate η
- Perceptrons are limited to linearly separable functions.

Activation Functions

- used to determine the output of each node of the neural network
  - Linear
  - Non-linear: most commonly used as it allows the model to generalize or adapt with variety of data

Backpropagation Algorithm

- most popular algorithm for learning ANNs
- t has similarities with the learning algorithm used in perceptron networks
- **Intuition**:
  - each unit is responsible for a certain fraction of the error in the output nodes to which it is connected

    - thus, the error is divided according to the weight of the connection between the respective hidden and output units, thus propagating the errors backwards
- Backpropagation computes the gradient in weight space of a feedforward neural network, with respect to a loss function
- **Algorithm**:
    - Initialize network weights (often small random values)
    - Do
        - For each example in training set
            - predict the output
            - calculate the prediction error by a loss function
            - compute δh for all the weights from hidden layer to output layer
            - compute δi for all the weights from input layer to hidden layer
            - **update network weights**
    - Until it converges
    - Return the Network
- **Stopping Criteria**:
    - maximum number of iterations
    - error based on the training set (when the error in the training set is below a certain limit.)
    - error based on a validation set (independent of the training set)(when the error on the validation set has reached a minimum)

Issues

Network Topology

- **The number of nodes in the hidden layer**
    - few nodes: underfitting
    - many nodes: overfitting
    - there are no criteria for defining the number of nodes in the hidden layer
- **Effect of learning rate** (sets the size of the steps to obtain the direction of maximum descendent)
    - a small learning rate has the effect of learning times higher
    - a high learning rate may lead to non-convergence

Generalization vs Specialization trade-off

- **Optimal number of hidden neurons**
    - too many hidden neurons: you get an overfit, training set is memorized, thus making the network useless on new data sets
    - not enough hidden neurons: network is unable to learn problem concept
- **Overtraining**: too much examples, the ANN memorizes the examples instead of the general idea

Some relevant hyperparameters

- **Network Structure**
  - number of layers
  - number of neurons in each layer
  - weights initialization
  - activation function
- **Training Algorithm**
  - learning rate
  - number of epochs
  - early stopping criterion
  - weight decay (a regularization on the network weights)

Some Tips

- Data should be standarized
- Missing values in input features may be represented as zeros, which do not influence the neural net training process.
- Use one-hot encoding, there are M output neurons (1 per class)
- For each case, the class with the highest probability value
- Initialize the weights with small random values $[-0.05, 0.05]$
- Shuffle the training set between epochs, i.e. change the sequence of the examples
- The learning rate must start with a high value that decreases progressively
- Train the network several times using different initialization of the weights

Pros

- Tolerance of noisy data
- Ability to classify patterns on which they have not been trained
- Successful on a wide range of real-world problems
- Algorithms are inherently parallel

Cons

- Long training times
- Resulting models are essentially black boxes

Introduction to Deep Learning

- Deep learning = Deep neural networks

Convolutional Neural Networks (CNN)

- Feedforward neural networks

- Neurons typically use the ReLU or sigmoid activation functions
- Weight multiplications are replaced by convolutions (filters)
- Change of paradigm: can be directly applied to the raw signal, without computing first ad hoc features
- Features are learnt automatically

Convolution: mathematical operation between 2 matrices

Properties

- Reduced amount of parameters to learn (local features)
- More efficient than dense multiplication
- Specifically thought for images or data with grid-like topology
- Convolutional layers are equivariant to translation
- Currently state-of-the-art in several tasks

< Go back