**Determination of Oxidation States in XES Data with Machine Learning Methods**

**-**

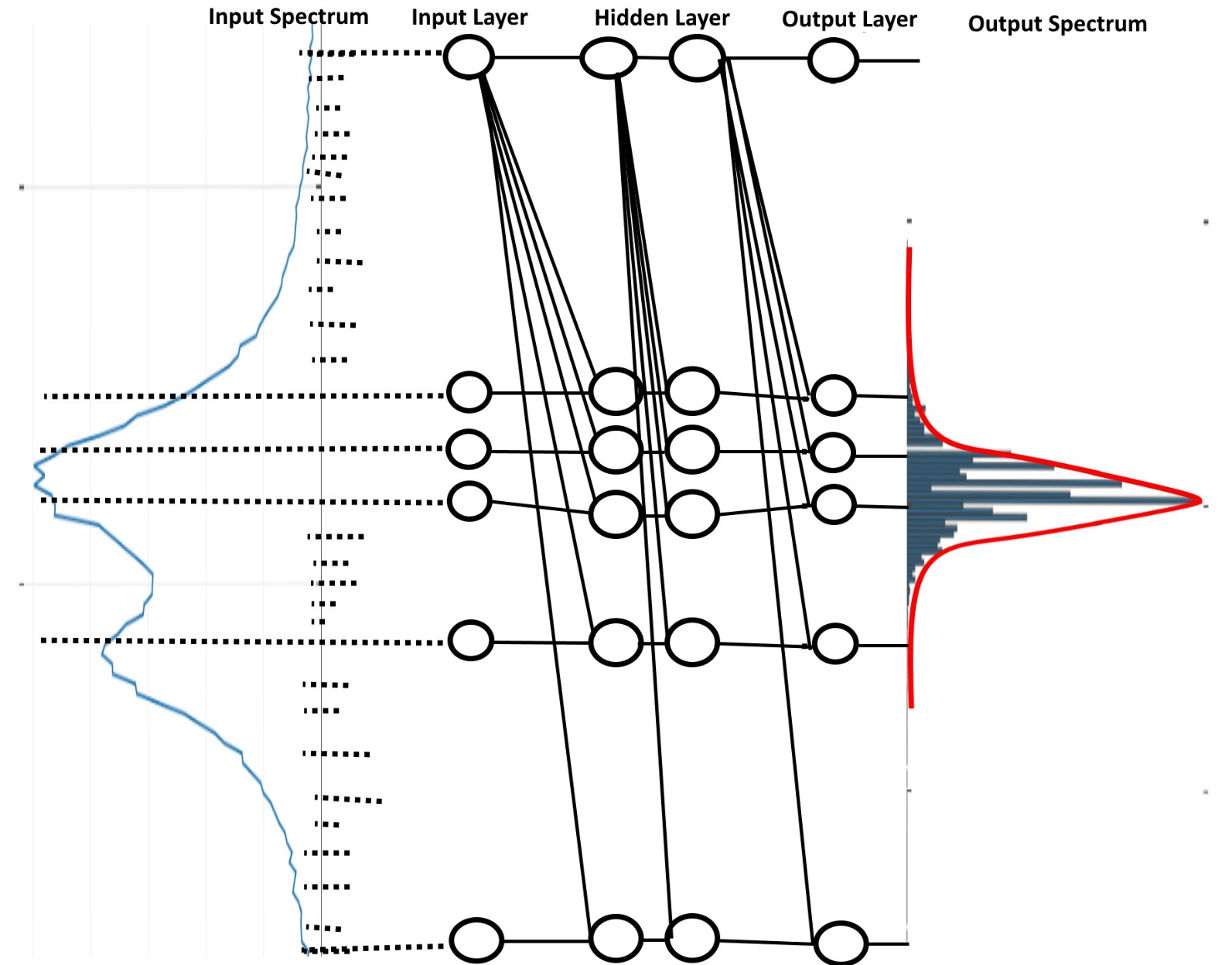
**Machine Learning Methods for Improved Interpretation of XES in the Analytical Determination of the Oxidation State Distribution Function**

Paragraphs for Figure 2: Schematic of ML approach

Over the last years the computation power of processors increased steadily. Machine learning algorithms developed in the 90s, who were thought to be too computational expensive to be useful at the time being, are now used in various fields, i.e. pattern recognition and event reconstruction. To examine the x-ray emission spectra, a supervised neural network was implemented in a *python* environment utilizing *scikit-learn’s MLPClassifier* library (ref). The training set was built by the randomly generated superposition of n \in [1,5] (<- math notation) oxidation states. Each state consists of a superposition of two Voigt-profiles where energy Kalpha1 was randomly chosen and the energy splitting between Kalpha1 and Kalpha2 was constant. The widths of the Lorentzian and Gaussian contribution to the Voigt-profiles were taken from (Ref to Wills paper ???) and left constant. To create input features in this multi-class classification problem, each spectrum was evaluated on a grid with 200 grid points between energies of 2008 eV and 2018 eV. Each training spectrum was tagged by a label randomly chosen from one of the Kalpha1 energies involved. The fully interconnected multilayer neural network (NN) consists of 200 input nodes, two hidden layers with 100 nodes each, the *logistic* activation function (ref), and an output layer which corresponds to an energy grid (“target classes”) ranging from 2012.80 eV to 2015.20 eV with a grid size of 0.01 eV. This gives 240 output nodes where the *Softmax* (ref) is applied as the output function. The output spectrum represents a probability distribution of Kalpha1 values in the input spectrum.

The generated sample library included 10e5 samples where 80% were used to train the NN, 10% were used as a development set to tune the hyperparameters, and 10% were used to evaluate the performance of the training procedure. Stochastic gradient descent with a batch size of 50, a momentum of 0.9, and a constant learning rate with a step size of 1e-4 was used to solve the *Cross-Entropy* (ref ??)loss function. The training was carried out by a desktop computer and took eight hours to complete the computation.

DRAF FOR FIGURE 2



**Figure 2**: Schematic of the neural network. By evaluating the input spectrum on a grid and creating labels the NN consisting of two hidden layers is trained. The output spectrum represents a probability distribution of possible Kalpha1 energies in the input spectrum.