Machine Learning Enabled Lineshape Analysis in Optical Two-Dimensional Coherent Spectroscopy

Michael Titze, Srikanth Namuduri, Shekhar Bhansali, Hebin Li

Department of Physics, Florida International University, Miami, Florida 33199, USA
Sandia National Laboratories, Albuquerque, New Mexico 87185, USA
Department of Electrocal and Computer Engineering, Florida International University, Miami, Florida 33199, USA
hebin.li@fiu.edu

Abstract: Although analytical solutions exist, the analysis of two-dimensional spectroscopy (2DCS) data can be tedious. A machine learning approach to analyzing 2DCS spectra is presented. We test the accuracy of the algorithm on simulated and experimental data.

Optical two-dimensional coherent spectroscopy (2DCS) is a powerful spectroscopic technique with applications ranging from revealing coupling in dilute atomic vapors over determination of relaxation dynamics in solid-state systems to the explanation of the energy conversion efficiency of photosynthetic complexes. A 2D spectrum is typically generated by focusing three laser pulses with adjustable time-delay ont a sample where a four-wave mixing signal is generated. Upon scanning any two time-delays, a 2D array is generated. Fourier-transformation of the time-domain spectrum yields a 2D frequency-domain map The linewidth in this spectrum is asssociated with the relaxation time. Hence, a critical task in analyzing 2D specta is the determination of the spectrums linewidth. Although closed-form analytic expressions have been developed for the ideal case of short pulses as well as for finite-width Gaussian and Lorentzian pusle shapes. However, manual fitting of 2D spectra to these anayltical lineshapes is a tedious task and practically impossible to perform during data acquisition. The use of our machine learning (ML) algorithm circumvents this issue by training on a large amount of training data, which can then be used for fast classification of unknown spectra. A major roadblock in many ML applications is the lack of sufficiently large sets of annotated training data.

Here, spectra of a single resonance resulting from the electronic transition of a two-level system are considered. Fig. 1 a shows a set of simulated spectra. The dashed diagonal line corresponds to absorption and emission at the same energy. The width along the diagonal and across the diagonal line are to be extracted using the neural network (NN) shown in Fig. 1 b. A spectrum of 128×128 pixels is read into the NN as a 2D array. The NN consists of two main stages, outlined in Fig. 1 b. The first stage is made up of three convolutional layers interspersed with two pooling layers. In a convolutional layer the input array is convoluted with a set of filters by computation of the dot product between the input and the filter. The output from a convolutional layer is a 2D map of the activation of the filter, referred to as a feature map. During training, the NN learns the filters that activate certain features in the input data. The output of the convolutional layer is then processed by pooling, effectively a way of downsampling the data. In this case, the max pooling function over 2×2 rectangular regions is used, which returns the maximum value of each region. Besides downsampling the data, this step also makes the NN more robust against noise in the input data. The first two convolutional layers are each followed by a pooling layer which reduces both data dimensions by half. After the third convolutional layer, the data is flattened into a 1D array as the input to a single dense layer.

The first dense layer consists of a fully connected network and is followed by a second dense layer which acts as the output layer consisting of only two nodes which give the homogeneous and inhomogeneous linewidths of the 2D spectrum.

In order to train the NN, an optimization of the filters in the convolutional layers is required. This optimization depends on the choice of loss function to minimize. The loss function is chosen as the root mean square error. If y_m is the measured linewidth and y_a is the actual width, the loss function is then defined as $L = \frac{1}{N}\sum_1^N (y_m - y_a)^2$. Optimizing the filters is mathematically equivalent to finding the global minimum of the loss function. Therefore, partial derivatives of the loss function are calculated with respect to the input variables and are used to update the filters of the convolutional layers.

The NN described in Fig. 1 b is implemented in Python using the Keras library which was run on Google Colaboratory. A total of 4096 spectra were simulated out of which 410 spectra were randomly chosen and set

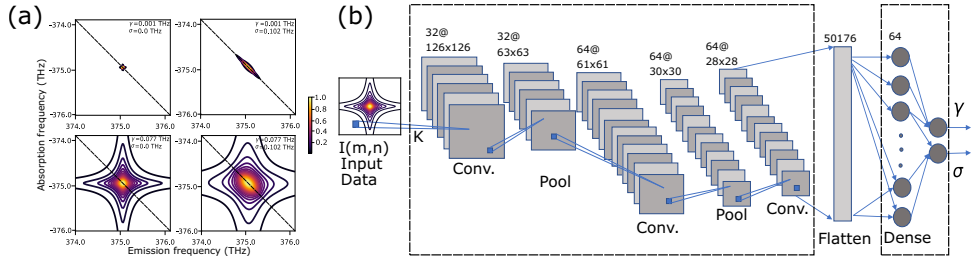

Fig. 1. (a) Four randomly selected 2D spectra used for training of the algorithm. (b) The schematic layout of the NN, including convolutional (Conv.), pooling (Pool), flatten, dense and output layer.

aside for testing. The remaining 3686 spectra were used for training. The NN was trained for 200 epochs with a learning rate of 10^{-5}. After training of the NN, the spectra not used during training were used to test the algorithm. Since these were simulated spectra, the actual linewidths were known and could easily be compared to the measured linewidths. The percent errors are shown in Fig. 2 a as a function of the homogeneous and inhomogeneous linewidth. Most of the measured linewidths are within a 2% error band for both homogeneous and inhomogeneous linewidth. The ratio between homogeneous and inhomogeneous linewidth strongly affects the data. Within the band of $0.25 < \sigma/\gamma < 2.5$, almost all datapoints show an error within the 2% error band. However, for $\sigma/\gamma < 0.25$, the error on the inhomogeneous linewidth gets large, up to 5% while for $\sigma/\gamma > 2.5$ the error on the homogeneous linewidth gets as large as 5%.

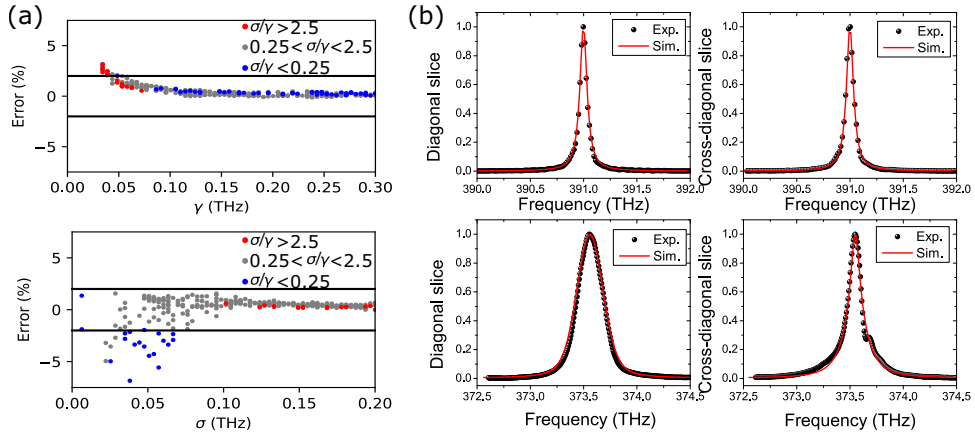

Fig. 2. (a) Percent error of the measured homogeneous and inhomogeneous linewidth. (b) Comparison between experimental and simulated slices of 2D data. The simulated slices use the parameters extracted by the NN.

Having validated the algorithm on the simulated spectra, the NN is applied to experimental 2D spectra. Two experimental spectra, from a potassium vapor and a GaAs quantum well, are used to test the NN. The two spectra were analyzed by the NN and the extracted linewidths are $\gamma = 0.038$ THz and $\sigma = 0.008$ THz for potassium vapor and $\gamma = 0.045$ THz and $\sigma = 0.103$ THz for the quantum well. These linewidths were used to simulate 2D spectra and diagonal and cross-diagonal cuts were taken from these simulated spectra and compared to the cut through the experimental spectra. The result is shown in Fig. 2 b, where the black dots are the slices from the experimental data and the red lines are slices from the simulated spectra using the extracted linewidth values. The simulated slices are in good agreement with the experimental slices, demonstrating the accuracy of the linewidth determination by the trained NN.

In summary, we have implemented a ML algorithm based on a convolutioal NN to analyze 2DCS spectra for linewidth information. After training with a set of simulated spectra with knkown linewidth values, the NN can analyze simulated 2D spectra as well as experimental 2D spectra to extract the homogeneous and inhomogeneous linewidths. The results are reliable and accurate and the ML algorithm can be used as an alternative to time-consuming conventional fitting of spectra.

References

1. M. Siemens et al., Opt. Exp. 18, 17, 17699 (2010)
2. I. Goodfellow et al., Deep Learning, MIT Press (2016)
3. S. Namuduri et al., arXiv:1911.11215