Deep learning for removal of non-resonant background in CARS hyperspectroscopy

A thesis submitted to the Committee on Graduate Studies

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in the Faculty of Art and Science

Trent University

Peterborough, Ontario, Canada

Materials Science M.Sc. Graduate Program

January 2023

Abstract

Deep learning for removal of non-resonant background in CARS hyperspectroscopy

George Aderopo Olaniyan

In this work, a deep learning approach proposed by Valensise *et al.* [3] for extracting Raman resonant spectra from measured broadband CARS spectra was explored to see how effective it is at removing NRB from our experimentally measured "spectral-focusing"-based approach to CARS. A large dataset of realistic simulated CARS spectra was used to train a model capable of performing this spectral retrieval task. The non-resonant background shape used in creating the simulated CARS spectra was altered, to mimic our experimentally measured NRB response. Two models were trained: one using the original approach (Specnet) and one using the updated NRB "Specnet Plus", and then tested their ability to retrieve the vibrationally resonant spectrum from simulated and measured CARS spectra. An error analysis was performed to compare the model's retrieval performance on two simulated CARS spectra. The modified model's mean squared error value was five and two times lower for the first and second simulated CARS spectra, respectively. Specnet Plus was found to be more effective at extracting the resonant signals. Finally, the NRB extraction abilities of both models are tested on two experimentally measured CARS hyperspectroscopy samples (starch and chitin), with the updated NRB model (Specnet Plus) outperforming the original Specnet model. These results suggest that tailoring the approach to reflect what we observe experimentally will improve our spectral analysis workflow and increase our imaging potential.

Keywords: non-resonant background; coherent anti-Stokes Raman Scattering (CARS) microscopy; deep learning; Raman retrieval; convolutional neural networks.

Preface to the thesis and Outline

Hyperspectral Coherent anti-Stokes Raman Scattering (CARS) is very influential in the world of material characterization. This technique provides non-invasive imaging and probes the chemical information of different molecules without labeling [1]. It was first implemented by Duncan *et al.* [2] and has evolved into a technique widely used to study the structural and chemical composition of various compounds. In CARS, two beams of laser light are typically used to excite the sample simultaneously; one beam is referred to as the pump beam, and the other is the Stokes beam [1]. Infamously, CARS suffers from the presence of a non-resonant background (NRB) that reshapes the observed CARS spectrum, degrading spectral analysis [3]. Finding a solution to this problem has been the primary motivation for this thesis project. Over the past few decades, several mathematical approaches have been employed to these ends, such as the Time Domain Kramers-Kronig method (TDKK) [4] and the maximum entropy method (MEM) [5], yet they all have their limitations when it comes to fully eliminating NRB signals [3]. Recently, machine learning has become a leading tool for data analysis and performing various applications, such as image classification for cancer detection [6]. Due to its recognition as a tool used for learning and classifying features from images, several machine learning approaches have already been developed to solve this problem in CARS. In this work, I explore a deep learning approach used by Valensise *et al.* [3] to remove these unwanted signals from measured multiplex or broadband CARS spectra and I study how best to integrate this method into our unique experimental approach to spectral-focusing CARS (SF-CARS) microscopy.

In this thesis, I report how I integrated this approach into out SF-CARS setup, the modification I made to the process, how the model performed with and without the change, and some future considerations to improve this work. Chapter 1 introduces the main concepts necessary to understand this work, such as Raman scattering, non-linear optical processes, and deep learning, and provides a background for the remaining chapters. Since this work revolves around improving spectral analysis in CARS, chapter 2 provides a more detailed review of the theory behind the CARS process, spectrum formation, and how the NRB signals distort the shape of the measured spectrum. The different variations to the basic CARS process and the different experimental implementations of hyperspectral CARS are then discussed. A detailed description of our spectral-focusing CARS setup is also presented.

The third chapter introduces convolutional neural networks and the Specnet framework, which is the deep learning approach in [3]. I outline the theory behind the process, and the results in applying the model to experimentally obtained hyperspectral data from our setup. Finally, I discuss the modification I made to the approach to better fit our setup.

Chapter 4 presents the results of using this deep learning approach with and without the modification on simulated and experimentally measured CARS spectra. The fifth chapter summarizes the thesis and offers suggestions for future improvements.

Acknowledgments

I'd like to express my deepest gratitude to everyone who has contributed, directly or indirectly, to the success of this project.

My biggest thank you goes to my supervisor, Dr. Aaron Slepkov, for believing in me and providing me with guidance, advice, and words of encouragement during my M.Sc. program, especially during the pandemic.

I'd also like to thank the Slepkov lab alumni, Jeremy Porquez and Ryan Cole, for teaching me how to use our laser setup, assisting with my transition into the lab, being available to answer all my questions, and their words of wisdom.

Many thanks also go to the many people who contributed to this work's foundation: Valensise *et al*. [3] for introducing the deep learning approach I explored for this work. Jeremy Porquez deserves special recognition for his preliminary work on adapting the published code and his assistance in resolving the numerous bugs I encountered while implementing the code.

I would also like express my gratitude to my supervisory committee members, Dr. Andrew Vruegdenhil and Dr. Franco Gaspri, for their insightful comments during my committee meeting.

Finally, I would like to thank my loving family back in Nigeria and Ireland for their continuous support throughout my academic journey. Being apart from them for the past few years hasn't been easy, but they are the main reason I persisted.

Thank you! E se gan ni! Go raibh maith agat!

Table of Contents

List of Figures

List of Tables

Chapter 1

Background Ideas: Raman Scattering, non-linear optical processes, and machine learning.

Machine learning and Raman spectroscopy are the two areas of research that form the foundation of this project. This chapter will provide the required foundation knowledge to comprehend how these two fields were combined in this work. Raman spectroscopy is a non-invasive and label-free technique used to probe the vibrational mode of a sample of interest [7]. Raman spectroscopy has been prominently used to perform a variety of applications in several fields, such as biology [8], medicine [9], and so on. Spontaneous Raman scattering or Raman effect, the phenomenon on which the Raman spectroscopy is based, will be introduced in Section 1.1. Although spontaneous Raman scattering has many benefits, it is relatively weak in most applications. Hence several enhanced Raman techniques have been developed with the introduction of new instrumentation, such as ultrafast lasers, to circumvent this drawback. [10]. These variants include surface-enhanced and non-linear Raman techniques. This work primarily deals with a non-linear Raman technique named Coherent Anti-Stokes Raman Scattering (CARS). Section 1.2 will discuss several non-linear optical processes and their relationship with Raman scattering, a linear optical process.

Machine learning is covered in the later section of this chapter. A “deep learning” strategy was specifically enlisted for this project, which is a subset of machine learning. Understanding deep learning as a concept is required to comprehend the complexity of the

approach employed in this work. Section 1.3 gives the reader a base knowledge of the relationship between machine learning and deep learning. It also introduces neural networks which are used to implement deep learning algorithms.

1.1 Raman Scattering

When a sample interacts with a light beam, there are two types of scattering that can occur. The first of which is known as Rayleigh scattering, an elastic process, in which energy is conserved, and the emerging light is of the same frequency ($E = E_0$) [11]. The other type is Raman scattering, an inelastic process consisting of a photon's absorption or emission and a phonon's emission or absorption. [11]. As described in Figure 1.1, when the emitted photon's energy is less than that of the incident photon ($E < E_0$), this is known as the Stokes process, whereas when it's greater than the energy of the incident photon ($E > E_0$), it is referred to as the anti-Stokes process [11].

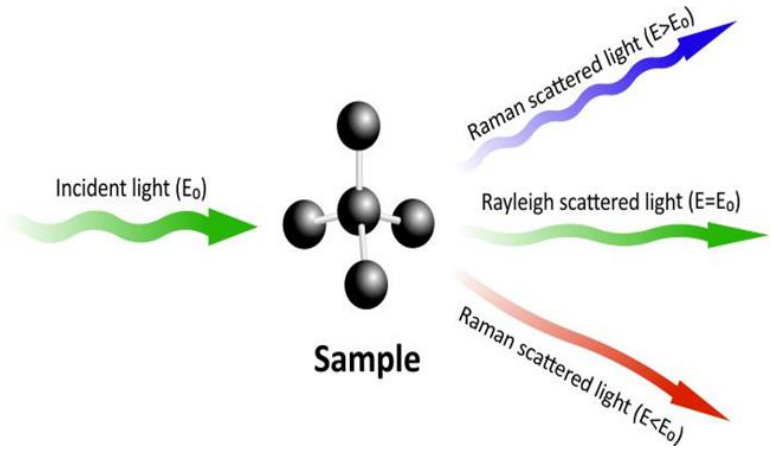

Figure 1.1: All three different types of light scattering. Rayleigh Scatter (green photon), Stokes Raman Scatter (red photon), anti-Stokes Raman Scatter (blue photon). The image is adapted from [12].

The energy difference between the two processes corresponds to the energy necessary to excite a specific vibration mode, as seen in Figure 1.2. When these scattered photons are detected, a Raman spectrum is created, with several bands representing the vibrational frequencies of various functional groups [13].

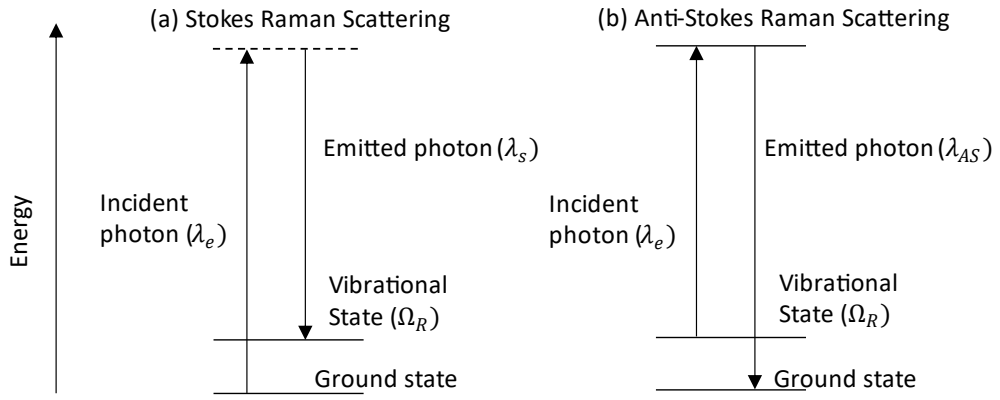

Figure 1.2: Jablonski energy diagram representations of Stokes and anti-Stokes Raman Scattering. (a) Stokes Raman Scattering. (b) Anti-Stokes Raman Scattering. The Stokes process leaves the molecule in a vibrationally-excited state. The anti-Stokes process starts with a vibrationally-excited state and returns the molecule to a lower excited state or the ground state.

When the emitted photon is shifted towards a longer wavelength, this is known as the Stokes shift. In contrast, the anti-Stokes shift when the photon is shifted toward a shorter wavelength. The change in wavelength for the Stokes and the anti-Stokes signals is expressed as:

$$\frac{1}{\lambda_S} = \frac{1}{\lambda_P} - \Omega_R \quad Stokes \tag{1.1}$$

$$\frac{1}{\lambda_{AS}} = \frac{1}{\lambda_P} + \Omega_R \quad anti - Stokes \tag{1.2}$$

where, λ_P, is the incident light's wavelength, λ_{AS}, is the anti-Stokes wavelength, λ_S is the Stokes wavelength and, Ω_R, is the frequency of the vibrational mode. As we know, each molecule has a distinct structure and set of vibration modes. Raman microscopy and other succeeding techniques developed via Raman scattering are used to probe the vibrational modes of each molecule in a sample of interest.

1.2 Non-linear optical processes: A brief review

This work focuses mainly on CARS microscopy, but we must consider several other non-linear optical phenomena. Non-linear optical processes occur when two or more photons interact simultaneously with a sample material [14]. As described in [15], the sample's induced (non-linear) polarization, *P*, is related to the electric field strength, *E*. This relationship is expressed as:

$$P = \epsilon_0\left[\chi^{(1)}E + \chi^{(2)}E^2 + \chi^{(3)}E^3 + \cdots + \chi^{(N)}E^N\right] \tag{1.3}$$

where ϵ_0 is the permittivity of free space, $\chi^{(N)}$ and E^N, are the nth-order (non-linear) susceptibility and corresponding electric field, respectively. The first term in equation 1.3 is the polarization contribution of linear optical processes (Raman scattering), and the higher-order polarization contributions are “non-linear” processes. The value of optical susceptibility decreases rapidly with increased order (i.e., $\chi^{(1)}$, is several orders of magnitude larger than $\chi^{(2)}$ and so on) [15]. However, the weaker effect from the higher-order contributions is overcome with the use of an ultrafast signal generation. With the use

of ultrafast lasers that can generate sufficiently high field strengths, the higher-order terms can become comparable to the linear term and thus the high-order term are measurable.

Second harmonic generation (SHG) and sum frequency generation (SFG) occur because of the second-order response described by, $\chi^{(2)}$, the second term in equation 1.3. They both occur when two photons interact with a sample material. Two-photon excitation fluorescence (TPEF), CARS, and third harmonic generation (THG) are due to the third-order response, $\chi^{(3)}$. These third-order non-linear processes are also referred to as four-wave mixing (FWM) processes. Four-wave mixing occurs when three incident laser fields with frequencies ω_1, ω_2, and ω_3, interacts with sample material, $\chi^{(3)}$, to generate signal field at frequency, ω_4 [16].

Non-linear optical processes are often integrated simultaneously into a multimodal microscopy system. Our multimodal system includes SHG, TPEF, and CARS. Figure 1.3 below shows the energy diagrams for SHG, TPEF, and CARS. Our multimodal setup is presented and further discussed in Chapter 2. The following subsections (1.2.1-1.2.3) briefly discuss these processes, their probing mechanism, and the type of molecule or material each technique can examine.

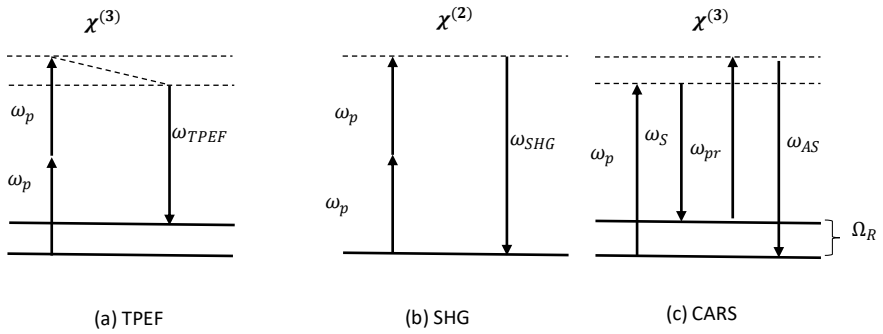

Figure 1.3: Jablonski energy diagram for (a) Two-Photon Excitation Fluorescence (TPEF), (b) Second Harmonic Generation (SHG), and (c) Coherent Anti-Stokes Raman Scattering (CARS). $\chi^{(2)}$, indicates the processes that occur due to the second-order response (SHG), whereas TPEF and CARS is due to the third-order response, $\chi^{(3)}$.

1.2.1 Two-Photon Excitation Fluorescence (TPEF)

In the microscopic implementation of multiphoton excitation, in which two or more photons induce an electronic transition from the ground state to the excited states via the virtual electronic states, the signal generation is proportional to the intensity of the excitation light squared [15]. The process described is shown in Figure 1.3 (a). Unlike one-photon excitation, where a single photon is used to excite a fluorescent molecule in the ultraviolet or visible range (400nm-500nm), in TPEF, the fluorescent molecule is excited in the infrared range (800nm-1000nm). Thus, the excitation is different, but the molecule fluorescence is the same. These allow for increased depth penetration and reduced photodamage of live cells and tissues [15]. This has led to several applications in the biomedical field, such as neuroscience [17], cellular and tissue imaging [18], and several other applications.

1.2.2 Second Harmonic Generation (SHG)

For the SHG process, two pump photons with a frequency, ω_p, interact with a target molecule to produce a beam precisely double the original frequency ($\omega_{SHG} = 2\omega_p$), i.e at half the wavelength. SHG can only occur in materials that exhibit non-centrosymmetric structures (i.e., those without a center of inversion symmetry) [14]. Examples of materials that exhibit this symmetry are collagen and pharmaceutical crystals [14]. However, it is worth noting that THG, which is due to the third-order response described by, $\chi^{(3)}$, or as an indirect two-step process involving SHG and SFG, can occur in materials that exhibit both centrosymmetric and non-centrosymmetric structures [15].

1.2.3 Coherent Anti-Stokes Raman Scattering (CARS)

As mentioned in the section preface, CARS microscopy is based on four-wave mixing process. In this process, as seen in Figure 1.3 (c), the pump, Stokes, and probe beams with frequencies, ω_p, ω_S, and ω_{pr}, respectively interact with the target sample to yield an anti-stoke signal with a frequency, ω_{AS}. This normally weak signal is amplified, when the frequency difference between the pump and Stokes beams matches a vibration mode (Ω_R) of the material. The sample of interest can thus be probed over various vibrational frequencies to achieve label-free imaging and spectroscopy. The CARS process is discussed in more detail in Chapter 2.

1.3 Machine Learning: A brief review

Artificial Intelligence (AI), arising from the use of machines to perform different tasks on their own, has been a practical approach to human learning and reasoning [19]. Early AI implementation focused on using a predefined set of rules provided by an expert to make predictions [20]. However, a less rigid structure is often needed due to the increased scale and amount of data. These constraints have led to the need for a more data-driven approach, such as *machine learning (ML)* and *data mining*. Machine learning, a subset of AI, has widespread use in research to learn from training datasets fed into the algorithm to predict outcomes in various applications, including text mining, spam detection, and image classification [21]. The different approaches to ML include Clustering, Bayesian networks, Deep Learning, and Decision Tree Learning [19]. This section will highlight Deep learning (DL), as it's the approach used. Figure 1.4 helps visualize how DL relates to machine learning (ML) and artificial intelligence (AI).

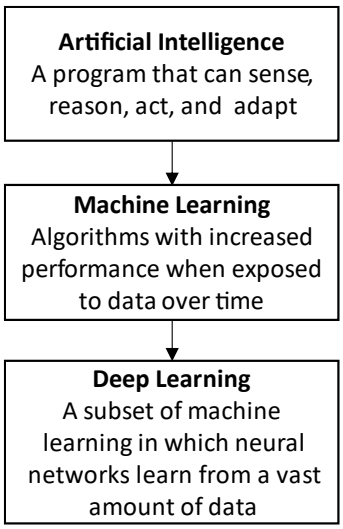

Figure 1.4: The deep learning family tree depicting the relationship between artificial intelligence, machine learning, and deep learning. Artificial Intelligence is the umbrella term for various computational methods of human reasoning and learning. One of these computational methods is machine learning, which focuses on using data to mimic human learning while also increasing learning accuracy. Deep learning is a branch of machine learning that employs artificial neural networks for data analysis and prediction.

1.3.1 Deep Learning:

The DL concept was introduced in 2006 [19] as a novel approach to machine learning. DL algorithms use numerous layers of perceptron, described in Figure 1.6, with each providing a different representation of the data fed to them. DL enables learning to be achieved in a single shot, by automating the learning and classifying different features for several types of datasets [21]. This is not the case for conventional ML techniques which require human intervention for the feature extraction step. Deep learning approaches are categorized into *supervised* and *unsupervised* learning. Supervised learning is a deep learning approach that

uses labeled data where the model input and output are both defined [22]. Labeling entails adding predefined tags, such as name, number, and color, to the raw datasets allowing the model to learn from an example. An example of this can be specifying whether an input image is that of a cat or dog. For unsupervised learning, the model learns by analyzing an unlabeled dataset and identifying hidden structures or similarities between each data point. The approach used for this work is **supervised learning** as further discussed in Chapter 3.

1.3.2 Neural Networks

Deep learning is designed using a multilayer algorithm known as a neural network. The building blocks of a neural network constitute the processing elements and nodes, whose functionality is based on the nervous system [23]. Each perceptron or neuron connects to another of its kind to form a **neural network or multi-layer perceptron,** and each has associated weights and thresholds [24]. When each node's output exceeds the specified threshold value in the layer, the node is activated, sending data to the next layer. Data is not passed to the next layer if this threshold constraint is not met. Figure 1.5 shows the constituent of each perceptron; a neural network unit. The perceptron comprises two steps: the first step calculates the weighted sum of the input functions, and the second is an activation function that transforms the output to a desired non-linear format before it is passed to the next layer in the network. The first step is mathematically expressed as [24]

$$\sum_{i=1}^{n} w_i x_i + b = w_1 x_1 + w_2 x_2 + \cdots + w_n x_n + b \tag{1.4}$$

where, x_1 through x_n, are the input data, denoted by the vector x on the left side of the equation; x_i, represent the i^{th} entry from the dataset; weights, w_1, through w_n can be denoted by the matrix, w_i; and b is the constant *bias term*.

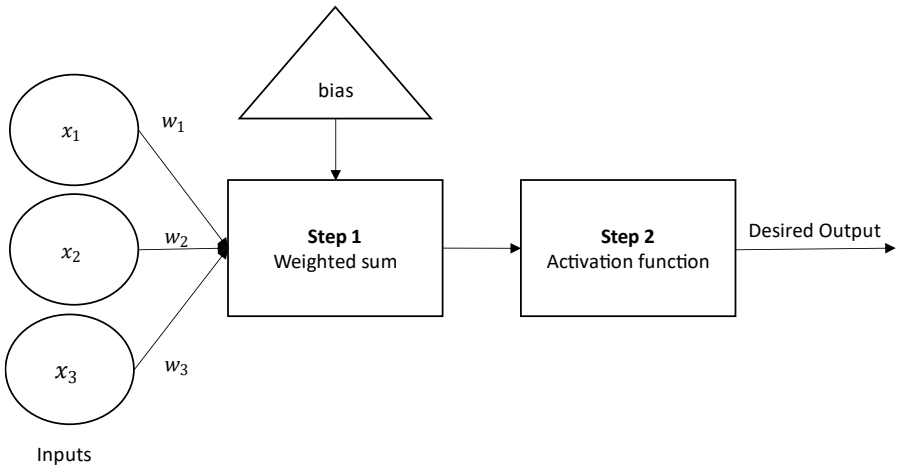

Figure 1.5: A perceptron; a single-layer neural network unit. The weighted sum is calculated in Step 1. In step 2, an activation function is used to transform the output to a desired non-linear format.

In the second step, the activation function helps map the summation result to the desired range [24]. Activation functions, including the sigmoid or logistic activation function, tanh activation function, rectified linear unit (ReLU) activation function, and so forth, are used for this non-linearity operation. The simplest neural network architecture is made up of three layers, commonly referred to as a "feed-forward neural network": an input layer, one

or more hidden layers, and an output layer [24]. Its inputs are processed in the forward direction; the input layer gathers the information, then passes it to the hidden layer for processing, and the output layer shows the processed results. This neural network is the primary form of other neural networks such as recurrent neural networks (RNNs) and convolutional neural networks (CNNs). CNNs in particular are used for applications that have to do with images due to their ability to learn from data with grid patterns, which makes it suitable for this work due to the nature of our data (hyperspectral images) [25]. The conventional CNN architecture will be further described in Chapter 3, along with how it was applied to this project.

Chapter 2

Hyperspectral Coherent Anti-Stokes Raman Scattering

2.1 Chapter Preface

This chapter provides insight into the origins of Coherent Anti-Stokes Raman Scattering (CARS) microscopy, the theory behind the process, spectrum formation, and how the non-resonant background (NRB) influences the shape of the CARS spectrum. The different variations in the basic CARS process and the different experimental implementations of hyperspectral CARS microscopy are mentioned. A schematic of our particular implementation of spectral-focusing CARS (SF-CARS) is presented. The chapter also presents an example of a hyperspectral image obtained from a sample of crystalline sucrose, highlight how the NRB distorts the observed CARS spectrum.

2.2 Introduction

CARS was first reported as a third-order non-linear optical process in 1965 by Terhune *et al.* [26] at Ford Motor Company. The name was not coined until a decade later, when it was used to study the chemical reaction between benzene and toluene by Begley *et al.* [27]. The first use of CARS as an imaging technique was in 1982 by Duncan *et al.* [2], who performed cellular imaging by temporally synchronizing the pulses to achieve phase-matching [2]. The introduction of femtosecond lasers led to the discovery of several other

non-linear processes and influenced further developments in CARS implementation. [1]. In 1999, Zumbusch *et al.* achieved CARS microscopy by changing the pump beam configuration to overlap co-linearly with the Stokes beam, using a femtosecond laser for close focusing to perform three-dimensional imaging [28]. The work by Zumbusch *et al.* gained CARS its recognition as a label-free alternative to existing fluorescence techniques for material characterization.

CARS offers a solution to the low-yield signal obtained in spontaneous Raman scattering, especially for microscopy [15]. Although it is a non-linear optical analog of Raman, due to its coherent nature, CARS processes can produce more than 10^5 times higher signal than the Raman process [15]. However, it also suffers drawbacks such as a square dependence on concentration, for example. So, it's not always better for dilute and trace sample characterization.

2.3 CARS basics

The CARS process involves interaction between three light beams denoted pump (ω_{pu}), probe (ω_{pr}), and Stokes (ω_s), to yield an anti-Stokes signal (ω_{as}). This signal is amplified when the frequency difference between the pump and Stokes beams matches a Raman-active vibrational mode (Ω_R) of the molecules in the sample [29].

As a third-order non-linear process, the electric field of the anti-Stokes signal and its corresponding angular frequency are given by [30]:

$$E_{as}(\omega_{as}) = \chi^{(3)} E_s(\omega_s) E_{pr}(\omega_{pr}) E_P(\omega_P) \quad (2.1)$$

$$\omega_{as} = \omega_P + \omega_{pr} - \omega_s \quad (2.2)$$

where E_s, E_{pr}, E_P , are the Stokes, probe, and pump fields, respectively; ω_s, ω_{pr}, ω_P, denotes the frequencies of the fields, respectively. $\chi^{(3)}$ denotes the third-order non-linear susceptibility.

For most CARS setups, the source of the pump and probe light is the same. This is referred to as a "degenerate" four-wave mixing (FWM), in which case equation 2.1 is expressed as:

$$E_{as} = \chi^{(3)} E_s {E^2}_P \quad (2.3)$$

For degenerate FWM, $\omega_P = \omega_{pr}$, and the corresponding angular frequency of the anti-Stokes is given by:

$$\omega_{as} = 2\omega_P - \omega_S \quad (2.4)$$

Theoretically, the anti-Stokes intensity for degenerate FWM will be proportional to the squared modulus of the non-linear susceptibility $|\chi^{(3)}|^2$ and is described as follows [15]:

$$I_{as} \propto I_P^2 I_S |\chi^{(3)}|^2 \tag{2.5}$$

where, I_P and I_S, are the pump and Stokes intensities, respectively. The non-linear susceptibility, $\chi^{(3)}$, is the sum of the non-resonant part, $\chi_{nr}^{(3)}$, commonly referred to as the non-resonant background, which appears due to off-resonance electronic contributions. The vibrationally-resonant part, $\chi_r^{(3)}$, which contains the chemical information of the sample of interest. The energy diagram for both the resonant and non-resonant FWM is depicted in Figure 2.1.

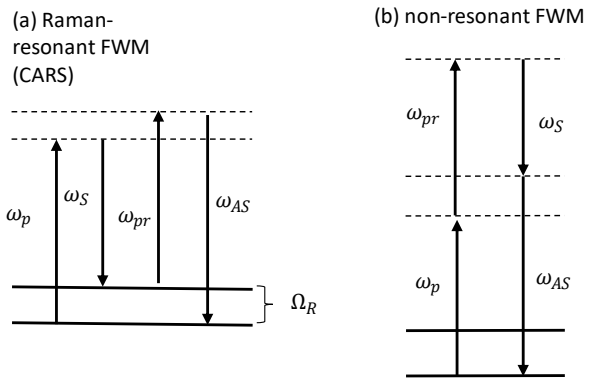

Figure 2.1: (a) Energy diagram representation for both Raman resonant (CARS) and non-resonant FWM (NRB) processes. This process involves the frequency mixing of three light sources, namely the Stokes, probe, and pump, with angular frequency ω_s, ω_{pr}, ω_P respectively, to produce an anti-Stokes signal with angular frequency ω_{as}. The anti-Stokes signal is amplified when the frequency difference between the pump and Stokes beam matches a vibrational mode of a molecule in a sample. (b) Energy diagram for electronic non-resonant FWM. The interaction between the pump and stokes beam within a molecule to generate a signal denoted anti-Stokes despite the fact no vibrational mode is probed.

The equation for the non-linear susceptibility, $\chi^{(3)}$, is simply [15]

$$\chi^{(3)} = \chi_{nr}^{(3)} + \chi_r^{(3)}. \tag{2.6}$$

The resonant part, $\chi_r^{(3)}$, is a complex function described as a Lorentzian function mirroring a spontaneous Raman line [31]. It is expressed as:

$$\chi_r{}^{(3)} = \frac{A_R}{\Omega_R - (\omega_P - \omega_s) + i\Gamma} \tag{2.7}$$

where A_R is the amplitude, or "oscillator strength", Ω_R, is the vibrational frequency, and Γ is the linewidth of the vibration resonance.

The anti-Stokes intensity for equation 2.5 can be expanded as:

$$\begin{aligned} I_{as} &\propto \left|\chi^{(3)}\right|^2 \\ &= \left|\chi_{nr}^{(3)} + \chi_r^{(3)}\right|^2 \\ &= |(\chi_r^{(3)})^2 + (\chi_{nr}^{(3)})^2 + 2\chi_r^{(3)}\chi_{nr}^{(3)}|. \end{aligned} \tag{2.8}$$

A simulated plot of equation 2.8 (shown in Figure 2.2) highlights how the combination of the resonant and non-resonant contributions distorts the shape of a CARS spectrum. This simulation considers the complex function presented in equation 2.7. The blue line represents the resonant part of the non-linear susceptibility, $(\chi_r^{(3)})^2$, containing the sample's chemical information. The dashed red line represents the purely non-resonant

contributions, $(\chi_{nr}^{(3)})^2$, which is assumed constant and is not frequency-dependent [32]. NRB is predominantly comprised of electronic signal contributions from other non-linear optical phenomena, that are less chemically specific [33].

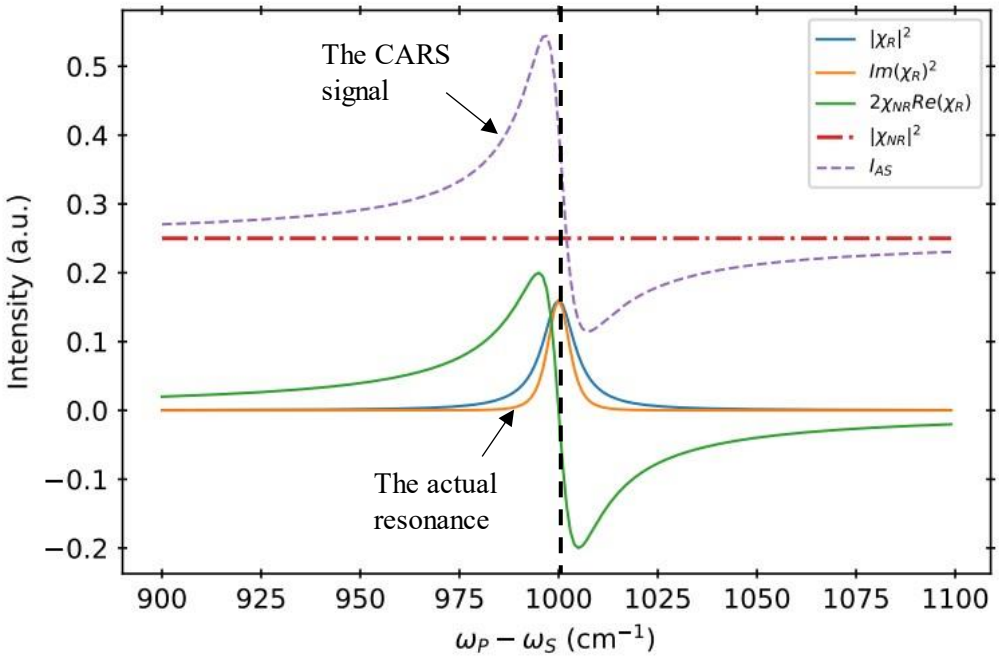

Figure 2.2: Constituents of a CARS spectrum. The vibrational resonance is set to 1000 cm^{-1}. $\chi_{nr}^{(3)}$ is set constant at 0.5, the amplitude is set to 2, the linewidth is set to 5, while both I_P and I_S, are set to 1. The third term, the mixing term, majorly influences the shape of the observed anti-Stokes signal. The dashed vertical line (black) was placed at the vibrational resonance (1000 cm^{-1}) to show the shifting of the peak in I_{AS}. The image and image description are adapted from [32].

The green line represents the third term, $2\chi_r^{(3)}\chi_{nr}^{(3)}$, an implication of the quadratic expansion of equation 2.5, a mixture of both the resonant and non-resonant parts. This term

is mainly responsible for reshaping the vibrational peaks into a dispersive line shape [31], as seen in Figure 2.2. Due to the complex nature of equation 2.8, this third term cannot be subtracted, making retrieval of the resonant term complex. The plot shown in Figure 2.2 is not an actual representation of experimentally measured CARS spectra but showcases how NRB reshapes the observed spectrum. Experimentally, other factors might also add to the complexity of experimental CARS spectra. Such factors include laser source noise, low target molecule concentration, symmetry of the functional group, and scattering losses. Several methods have been implemented to remove NRB from measured CARS spectra, leaving it with resonant parts. Clearly, simple subtraction of the $(\chi_{nr}^{(3)})^2$ is not enough because of the $2\chi_r^{(3)}\chi_{nr}^{(3)}$ term. Mathematical methods of spectral retrieval such as methods such as the time domain Kramers-Kronig method (TDKK), and maximum entropy method (MEM) have their limitations when it comes to performing this task [3], [6]. The next chapter discusses a deep learning approach's practicality in removing NRB from measured hyperspectroscopy.

2.4 Variations of the primary CARS method

Even though the focus of this work is removing non-resonant signal from CARS signals collected in the forward direction, I still want to draw attention to other CARS variations because our lab has recently explored these other approaches, and thus they may benefit from the DL approach described later in Chapter 3. The basic CARS configuration, also known as forward-detected (F-CARS), occurs when the resonant and non-resonant signals propagate in the same direction. Different variations include backward-detected or “epi-

detected" CARS (epi-CARS), and polarization CARS (P-CARS). epi-CARS is implemented by detecting backward propagating resonant signals using dichroic mirrors [34]. Compared to F-CARS, epi-CARS is extremely sensitive to the sample's size and shape since epi-signals can be greatly suppressed by destructive interference. [35]. However, epi-CARS exhibits lower NRB and can be used to detect small scatters in the sample's background, so it is advantageous to simultaneously detect both the forward and epi-scattered signals.

P-CARS works by exploiting the polarization difference between the resonant and non-resonant signals before detection [34]. P-CARS considerably enhances CARS sensitivity of weak Raman resonances by suppressing the background observed in the transparent media [36], [37]. In this method, the pump and Stokes beams are purposefully polarized in different planes. A significant CARS signal is produced by controlling the laser polarizations and inserting a polarization analyzer before the detector, to extinguish the non-resonant signal and pass a portion of the vibrationally resonant signal. While P-CARS is known to reduce the influence of NRB, this comes at the cost of the observed CARS signal intensity [37].

2.5 Hyperspectral CARS Implementations

Historically, the first implementation of CARS microscopy employed a narrowband pump and Stokes beam to excite a single Raman mode at a time [1], [28], [38]. This approach is most commonly implemented using a picoseconds optical parametric oscillator (ps-OPO). A primary laser output is split into two beams, one of which serves as the Stokes beam and

the other for synchronously pumping an ps-OPO to generate the pump beam [39]. This scheme allows for high-speed imaging, but it does not discriminate against molecules having overlapped Raman modes, which is a disadvantage in spectral analysis. A multi-color approach to this scheme gives way to “hyperspectral imaging” via wavelength sweeping [39].

Hyperspectral imaging provides a spectral profile for each pixel [38]. As seen in Figure 2.3 (a), wavelength sweeping uses two narrowband beams and continuously tunes the frequency of one of the beams over a range to perform hyperspectral imaging [38]. Hyperspectral imaging is also achieved in two other implementations, broadband CARS (B-CARS) and spectral-focusing CARS (SF-CARS). The broadband scheme utilizes a narrowband pump beam and broadband pump beam to excite all the sample's vibrational modes simultaneously (Figure 2.3 (b)). The CARS signals are collected by an array detector, such as a cooled charge-coupled device (CCD) acting as a spectrometer [40]. By contrast, the spectral focusing scheme (SF-CARS) involves spatially overlapping a pump beam with a portion of supercontinuum Stokes, wherein the frequency difference corresponds to a single vibrational mode made possible through chirp-matching (Figure 2.3 (c)) [41]. Chirp-matching is the idea of matching the frequency vs time dependence of both beams (pump and Stokes) to focus their frequency difference at a single vibrational resonance [41]. Imaging is done by scanning the overlap of both chirped beams at different frequencies, by delaying one pulse with respect to the other and collecting the observed signals with a single detector, such as a photomultiplier tube (PMT). The last two schemes (B-CARS and SF-CARS) are both plagued by NRB which often dominates the weaker resonant signal of interest [40]. The deep learning approach described later in this thesis

was originally employed to solve this problem for B-CARS, the approach was modified to perform the same task on our one-of-a-kind spectral focusing scheme. Our spectral focusing scheme used to perform hyperspectral CARS is introduced in section 2.6.

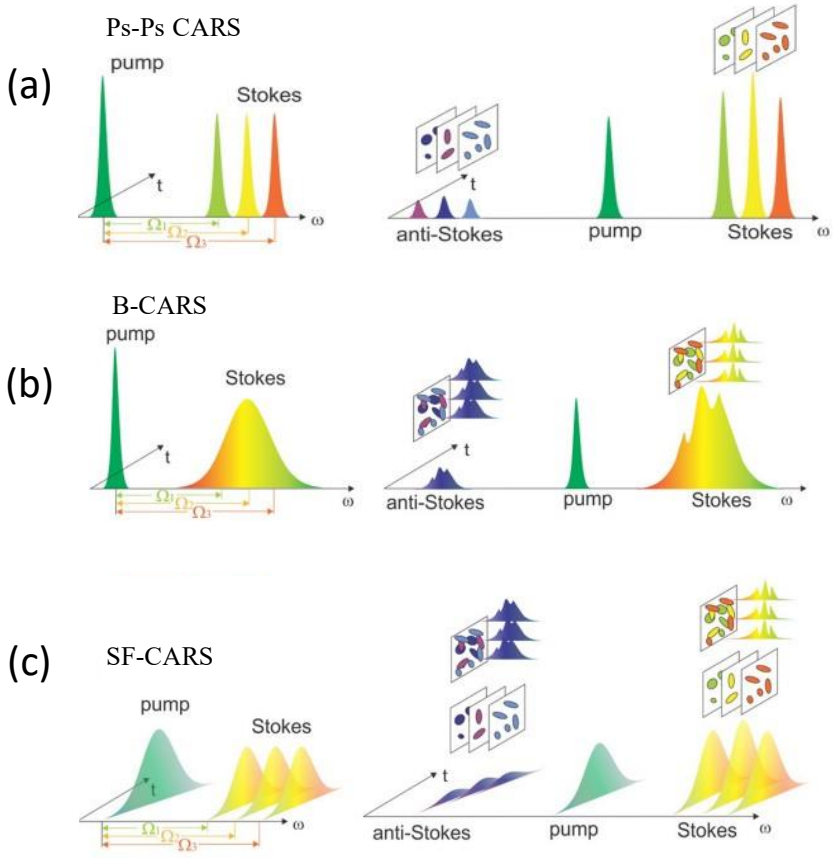

Figure 2.3: Different schemes used to implement hyperspectral CARS. (a) The single frequency scheme is based on the frequency-tuning of the narrowband pump and Stokes beam. (b) The broadband scheme is achieved by simultaneously exciting all the vibrational modes using a narrowband beam and broadband Stokes beam. (c) The spectral focusing scheme (SF-CARS) is based on spatially overlapping a narrowband or broadband pump with a portion of supercontinuum Stokes where the difference corresponds to a vibrational mode; this scheme is achieved by chirp matching the pulses. The image is adapted from [38].

2.6 The Spectral-Focusing CARS Scheme

A scheme for our SF-CARS setup is shown in Figure 2.3. As described in both [32] and [37], a Ti: Sapphire laser (Spectra-Physics Tsunami) generates an 800 nm beam with a tunable pulse duration ranging from 70 femtoseconds to 200 femtoseconds. The laser output is split into two paths: One beam serves as the degenerate pump and probe beam, and the other path is used for the Stokes beam. The first beam is sent to a computer-controlled optical delay stage (Thorlabs DDS220) before recombining with the Stokes. The delay stage controls the temporal overlap between the pump and the Stokes beam [41]. The second path is coupled into a commercial photonic crystal fiber (PCF) (FemtoWhite-CARS, NKT Photonics) using a $40\times$ objective lens. The PCF can generate a supercontinuum spanning 550 nm to 1200 nm, with the region >820 nm used for the Stokes beam [41]. A Mitutoyo (M Plan NIR $50\times$) long-working-distance objective is used to collimate the Stokes beam. The end-to-end coupling efficiency of the PCF is approximately 32%. The pump and Stokes beams are also co-linearly polarized by placing a half-wave plate (HWP) in each beam path before they are recombined using an angled dichroic mirror (Chroma T8101pxr) acting as an effective 840 nm long-pass filter (LPF). A linear polarizer is added before the entrance to the microscope to make sure the beams are linearly co-polarized. The Stokes and the pump beams are temporally dispersed and chirp-matched by a 6 cm and 10 cm of S-NPH2 glass, respectively, to implement spectral focusing.

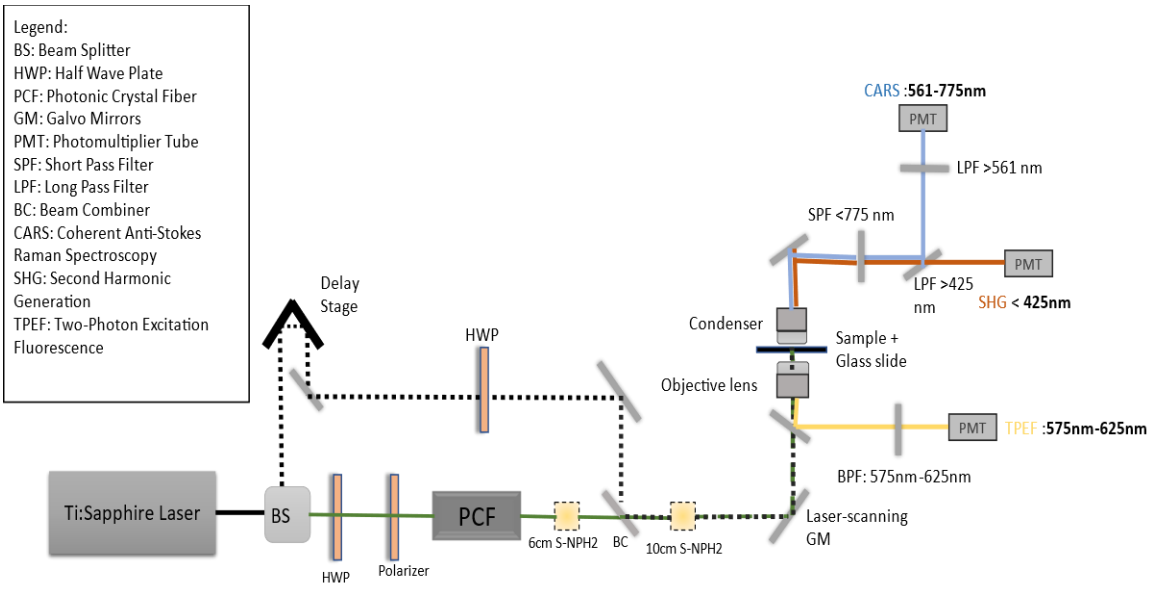

Figure 2.4: The schematics for the multimodal SF-CARS microscopy setup used in this work. A Ti: Sapphire laser generates an 800 nm beam with variable pulse duration from 70 to 200 femtoseconds. The laser output is split using a beam splitter into a Stokes and pump beam. The first beam is coupled into a FemtoWhite-CARS (NKT Photonics) module to generate a supercontinuum spanning 550 nm to 1200 nm, with the region >820 nm used as the Stokes beam. The delay stage controls the overlap between the pump and the Stokes beam. A 6cm block and a 10 cm block of S-NPH2 glass are used to disperse and chirp-match the Stokes to the pump beam, respectively. Both beams are recombined using an angled dichroic mirror acting as an 840 nm long pass filter. A linear polarizer is added before the entrance of the microscope to ensure both beams are co-linearly polarized; the beams are then sent into a laser scanning microscope. The CARS and SHG signals are detected in the forward direction, separated using a 425 nm LPF, and detected by different PMTs. TPEF are collected in the epi-direction and detected by a PMT. A series of short-pass, long-pass, and broadband filters are used to eliminate undesired signals. Computer

control, implemented via Python, is integrated into the system to manage and synchronize various parts of the scheme, as well as for data acquisition.

The microscope is a modified Olympus IX73 inverted laser-scanning microscope with a 1.15 NA 40 × objective (Olympus UAPON40XW340), a Thorlabs scanning galvo system, and a computerized sample stage [37]. CARS and SHG signals are detected in the forward direction and are wavelength-separated using a dichroic filter (Chroma T8101pxr), acting as a 425 nm long pass filter, with each signal sent to separate detectors [32]. Hamamatsu H10723-01 PMTs is used to detect the SHG, while a Hamamatsu H10723-20 PMT is used to detect the CARS signals [37]. A series of filters (short pass, long pass, and broadband) are used to eradicate unwanted signals, as shown in Figure 2.4. The filtering system can also be configured to allow for epi-CARS detection. A custom Phyton program is used for instrumentation control of vital parts such as the delay stage, shutters, detectors, galvanometer mirrors, and the microscope [32]. This program also produces an image stack from frame-by-frame hyperspectral data acquired at different frequencies.

2.7 CARS hyperspectroscopy:

This section presents the findings from a CARS hyperspectroscopy experiment on a sucrose sample using our setup (see Figure 2.5). The pump power was set to 75 mW, and the PCF input was set at 150mW for supercontinuum generation. The CARS hyperspectral stack consists of 250×250 pixel images at 698 spectral data points collected at different individual frequencies; spectral filtering and the Stokes wavelength limit our detection of CARS signals to a maximum range around 3600 cm^{-1}.

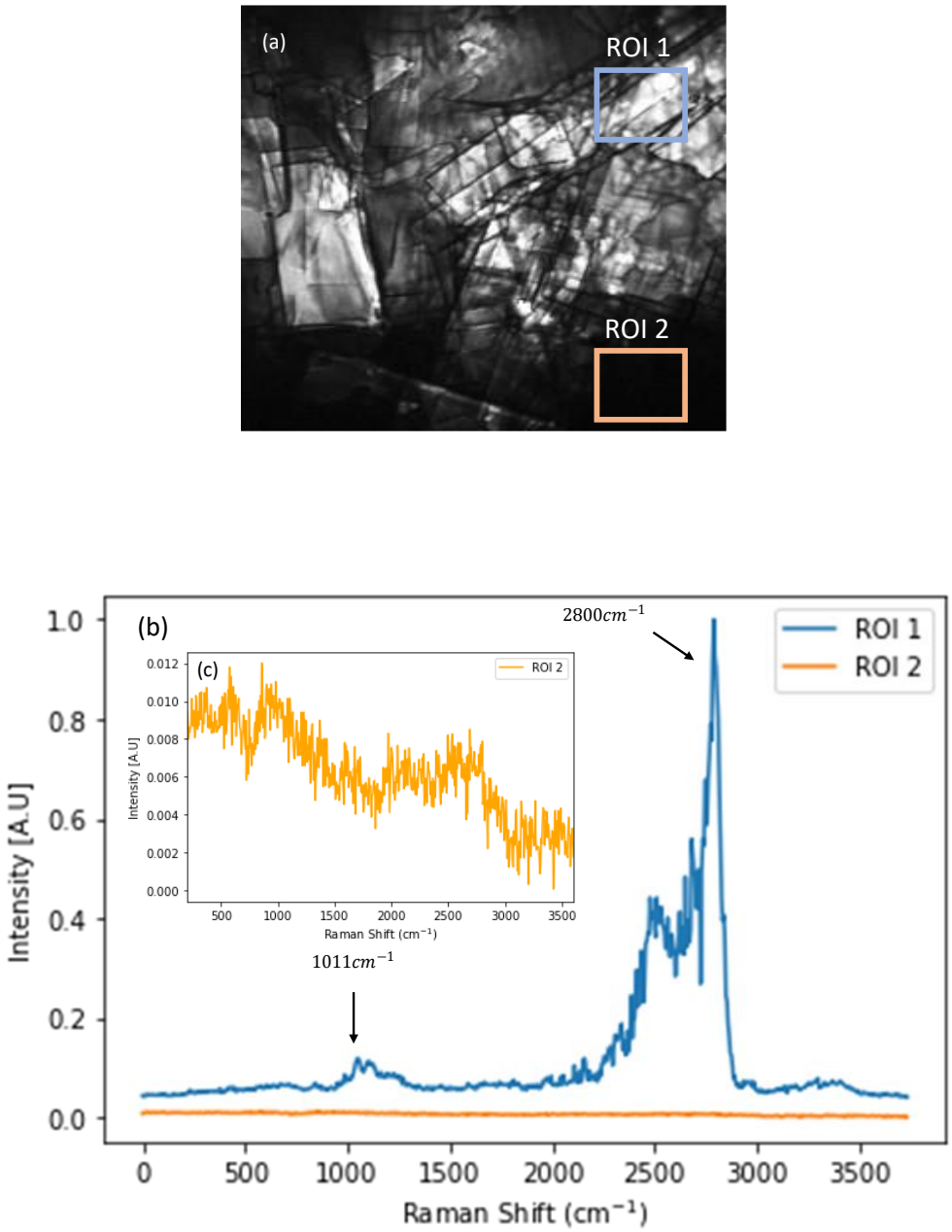

Figure 2.5: Broadband hyperspectroscopy for a crystalline sucrose sample. (a) A 250 × 250 pixel greyscale CARS image generated with an average of 11 frames centered at around $2800cm^{-1}$. (b) CARS spectra observed at two regions of interest in (a). (c) A zoomed-in representation of ROI 2 is displayed in the spectrum to show the NRB structure. ROI 1: Spectra observed from a region with a high sample concentration; ROI 2: Spectra

observed from a region containing the coverslip to represent the observed non-resonant background. The pump power was set to 75 mW, and the PCF input for supercontinuum generation was set to 150 mW. This characteristic vibrational signals for sucrose (at 1011 cm^{-1} and 2800 cm^{-1}) are shown. Other unresolved peaks and signals are a combination of noise and non-resonant background.

The 250×250 pixel image shown in Figure 2.5 (a) was generated as an average of 11 frames centered at around 2800cm^{-1}. (b) the CARS spectra presented were obtained from two regions of interest. ROI 1: a region with a high concentration of the sucrose sample; ROI 2: a region with just microscope cover slip. In the fingerprint region, spanning 600 cm^{-1} to 1400 cm^{-1}, the only strong characteristic resonance for sucrose is located at around 847 cm^{-1} [42]. Looking at the spectra presented in Figure 2.5 (b), in ROI 1, this resonance was not detected, suggesting limited Stokes power at the corresponding wavelengths [32]. However, ROI 1 shows a weak resonance at around 1011cm^{-1}, which can be attributed to known C-O stretching in sugars [43]. A strong resonance was detected at around 2800cm^{-1} arising from the strong C-H vibrations in sucrose. ROI 2 highlights the NRB contributions to the sucrose spectrum in ROI 1. According to the simulation Figure 2.2, the non-resonant contribution is relatively constant, and this is observed from the experimental results in ROI 2. The heavy distortions in the silent region, from 2000 cm^{-1} to 2500 cm^{-1} [3] , observed in ROI 1 spectra is attributed to the contribution from the mixing term, as seen in equation 2.5. As discussed in section 2.3, the resonant and non-resonant contribution mix complicates the spectrum observed, as seen in Figure 2.5 (b). Several methods have been used to reduce this complication. In this work, a deep learning

approach, a convolutional neural network (CNN) is employed to achieve this, and the results are presented in the next chapter.

Chapter 3

The Specnet Framework: A convolutional neural network

3.1 Chapter Preface

This chapter provides insight into convolutional neural networks (CNNs). Section 3.2 will discuss why CNNs work best for image-related learning and how their architecture supports these tasks. Sections 3.3- 3.6 will introduce the Specnet framework, a proposed deep learning model used for Raman retrieval from experimental B-CARS images and spectra [3], how the framework was implemented, and the modification made to the approach.

3.2 Convolutional Neural Networks (CNNs)

CNNs are used to process data with grid patterns, such as images, to learn spatial hierarchies by assigning weights and biases for classifications down the line [44]. The core components of a CNN are the convolutional, pooling, and fully connected layers. CNN is divided into two phases: The first phase comprises the convolution and the pooling layer. This phase is where feature extraction from the input is carried out. At this phase, the raw input is transformed into smaller relevant features for processing while preserving the original information. In the second phase, the fully connected layer maps and classifies the

extracted features to predict an output. Figure 3.1 provides a visual representation of the basic CNN architecture.

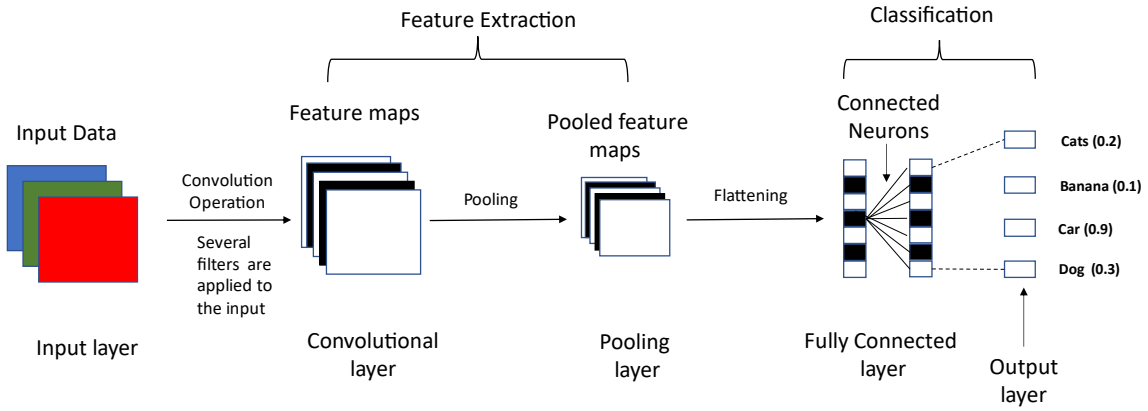

Figure 3.1: A visual representation of a convolutional neural network. The neural network can be primarily divided into two phases namely the feature extraction phase and the classification phase. The feature extraction phase comprises the convolutional and pooling layer. The raw input from the **input layer** is transformed into smaller relevant **feature maps** for processing while preserving the original information at the **convolutional layer**. The main function of the **pooling layer** is to reduce the dimensions of the extracted feature maps to increase computational speed. The output from the pooling layer (**the pooled feature maps**) is converted to 1-dimensional linear vectors via a **flattening step** before they are sent to the fully connected layer. The **fully connected layer** is made of several connected neurons that compile the vectors with similarly distinct features for classification and output a confidence score between 0 and 1 indicating the if the input belongs to a particular label (the highest confidence score was assigned to the "car" label which is the input image used in **Figure 3.2**).

3.2.1 Input Layer

CNN inputs are usually either an image or a video file which are a collection of pixels stored in arrays [44]. For grayscale images, the pixel values are stored in a one-dimensional array with integers that range from 0 (dark or black shade) to 255 (light or white shade), whereas colored images are stored in three-dimensional arrays because they are generated from a combination of the three primary colors: red, green, and blue. Each value in the arrays corresponds to the intensity of each color in each pixel. They also range from 0 to 255 depending on the color's intensity. For computational applications, the images are initially converted to their array form before processing using NumPy (a Python library). A visual example of this conversion is shown in Figure 3.2 below and this is used as a representation for CNN architecture in **Figure 3.1**. This layer is followed by the feature extraction phase where the initial raw input read as arrays are converted into feature maps.

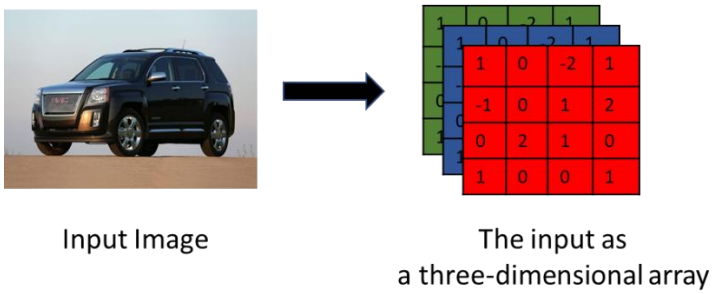

Figure 3.2: A colored image converted to a three-dimensional array for processing. Images are a collection of pixels stored in arrays. Each pixel in a colored image is made up of a combination and intensities of three primary colors which are stored in a three-dimensional array.

3.2.2 Convolution Layer

The convolutional layer is the most important part of any CNN architecture. Convolution entails using several learnable **filters** applied as weights (see Figure 1.6) to reduce the dimensionality of the raw input while preserving the original information. Each filtering operation produces an activation or a convolved feature and using the filters repeatedly throughout the input dimension produces a map of activations known as a **feature map** [45]. A typical feature map is just an amalgamation of distinct features from the input—for example, a group of pixels containing the tires might make up a feature map from the input image in Figure 3.2. To better understand how this convolution operation occurs, Figure 3.3 illustrates how a single filter is applied in a rolling manner across the dimensionality of the input. The figure shows preliminary calculation at three different steps for a 4×4 grayscale image (input) with a 2×2 weighted filter resulting in a 3×3 feature map. Additionally, it shows the final feature map once the filter has been used throughout the entire image.

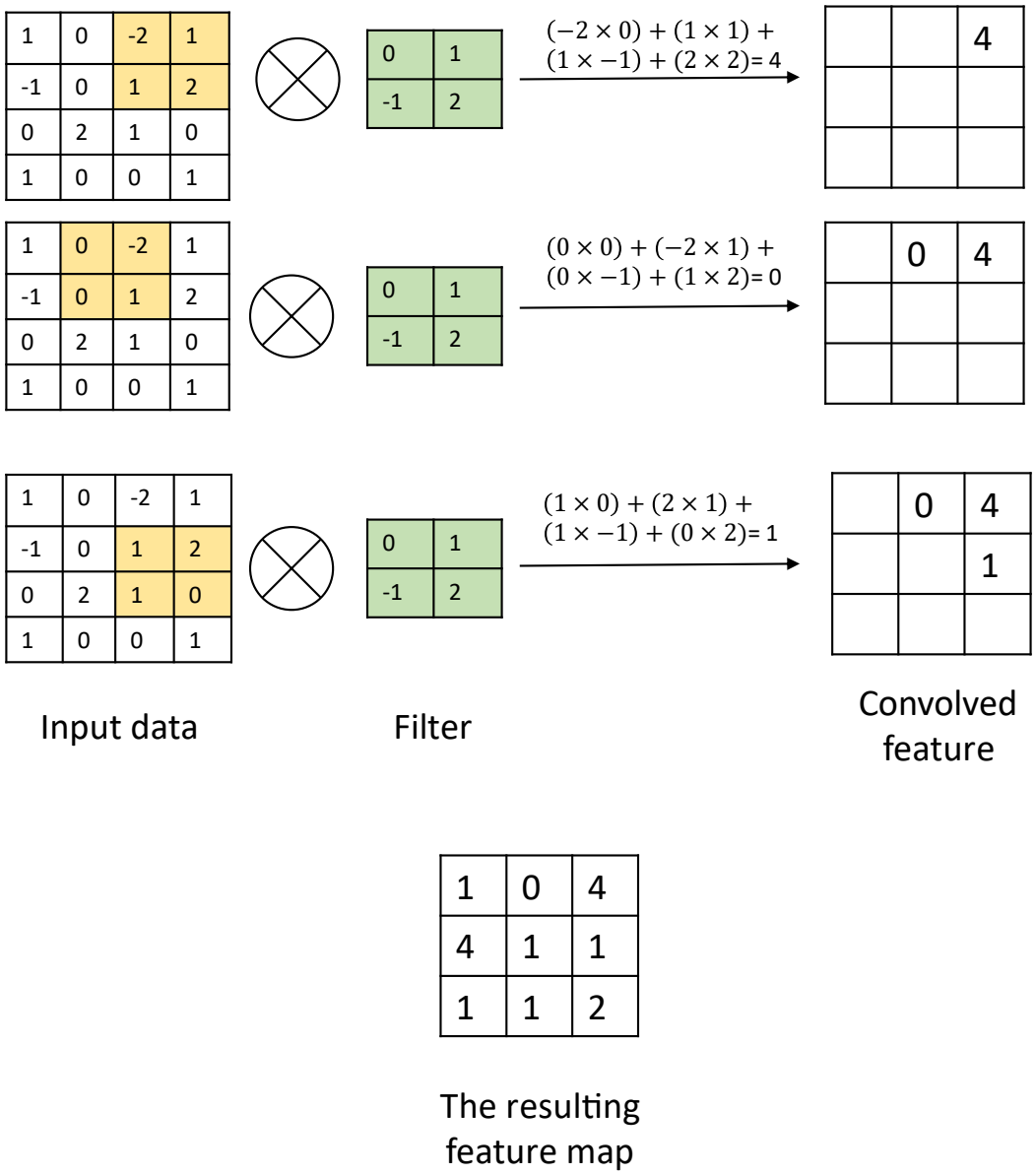

Figure 3.3: A representation of a convolution operation. For this convolutional operation, a 2×2 weighted filter (green) is applied in a rolling manner across the 4×4 grayscale input (yellow) to yield a 3×3 feature map (white). The calculations and results for three convolved features are shown. The complete 3×3 feature map which will be layer output is also shown.

As seen in Figure 3.3, a dot product is performed with the elements of the input data every time the filter is applied at each stage. These dot product results make up the feature map volume, which is the output of the convolutional layer [45]. Since the output depends on the applied filter, the number of features extracted at each layer can be controlled using different filter sizes. In the case of colored images, this operation is performed for the different colored arrays (red, green, blue).

3.2.3 Pooling Layer

The main responsibility of this layer is to reduce the size of the preceding layer's output (feature maps) while keeping most of its dominant features [21]. The reason for having this layer in the architecture is to further reduce the number of parameters needed, and in turn reduces, the computational complexity of the model [45]. Several pooling methods are used for this operation; they include tree pooling, gated pooling, average pooling, minimum pooling, and maximum pooling [21]. Figure 3.4 demonstrates how the two commonly used pooling methods, maximum and average pooling, perform this operation.

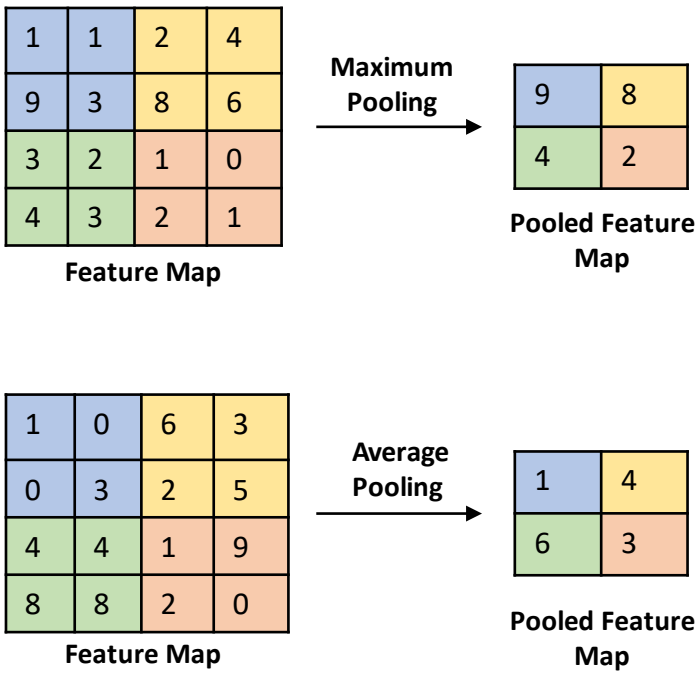

Figure 3.4: A demonstration of maximum and average pooling. A pooled feature map is created by pooling values from a segment of the feature maps, which reduces their dimensionality for better computational performance. Average pooling returns the mean value in each segment and maximum pooling returns the highest value in each segment.

Each pooling operation only takes place at a different segment of the input array, as opposed to the filters, which are performed in a rolling fashion in the convolutional layer. As seen in Figure 3.4, maximum pooling takes the maximum value at each portion to form a **pooled feature map**, whereas average pooling just restores the average value for each portion. The feature extraction phase is concluded once the output from the pooling layer (the pooled feature maps) is sent to the subsequent layer for classification.

3.2.4 The Fully Connected Layer (FC)

The next phase in the architecture, classification, is carried out at the fully connected layer as seen in Figure 3.1. Before data from the preceding layer (the pooled feature maps) is sent into the fully connected layer, the output matrix is converted to 1-dimensional linear vectors (This step is called **"flattening"**). This is then passed to the fully connected layer which is made up of neurons connected to all neurons from the previous layer. The purpose of these connections is to compile all the vectors with similar features. Weights are assigned to the complied vectors to quantify the presence of distinct features and predict the correct label. Figure 3.1 illustrates the layer's output, which is a set of confidence scores (numbers ranging from 0 to 1) that indicate the likelihood that the input belongs to a particular label (in this case, the confidence score was given to the car label which is the images used in Figure 3.2).

3.2.5 Activation Functions

As highlighted in Chapter 1, the later component of each perceptron in a neural network is an activation function. The major role of these functions is to predict an output using the weighted sum of its input. This is accomplished by having the function establish a threshold value. If this value is not exceeded, the information in the input is not sent to the next layer. However, if the value is exceeded, the node (**perceptron input**) is activated, and information is allowed to pass through. Practically, these functions are used to determine the non-linear relation between the input and output. The three most commonly used are the tanh, sigmoid, and rectified line unit activation functions (ReLU) but several other

functions can be used to perform this operation. A brief description of these three functions is as follows:

Tanh: This function accepts real values as input, and its output ranges from -1 and 1 [21]. It's mathematically expressed as:

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{3.1}$$

Sigmoid: This activation function only accepts real values as input, and its output ranges from zero to one [21]. The sigmoid activation function is expressed as:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{3.2}$$

ReLU: This is the most popular function for implementing CNNs, and the function employed in this work. Equation 3.3 shows the rectification at the bottom because $f(x)$ is zero when x (input in our case) is less than zero [21]. At the same time, $f(x)$ equals x when x is greater than zero. Any input with a negative weighted total is zeroed out by the function because the threshold is set to 0.

$$f(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases} \tag{3.3}$$

In a basic CNN architecture (Figure 3.1), an activation function can be added to the end of each layer to add non-linearity to the network. In the case of this work, a RELU function was included at end of each layer in our neural network.

3.3 The Specnet Approach

In this work, we employed a deep learning approach designed and published by Valensise *et al.* [3] to resolve the NRB limitations that come with broadband-rich CARS schemes. The approach was used to retrieve the resonant part of the non-linear susceptibility, $\chi_r^{(3)}(\omega)$, directly from a measured B-CARS spectrum, without the need for external measurements or complex processing [3]. The relationship between CARS and spontaneous Raman scattering, is [15]:

$$I(\omega)_{Raman} \propto \chi_r^{(3)}(\omega), \tag{3.4}$$

where, $I(\omega)_{Raman}$, is the measured Raman intensity and, $\chi_r^{(3)}(\omega)$, is the resonant contribution to CARS. This relationship makes it possible to compare these two techniques directly when performing spectral analysis after the non-resonant has been removed.

The following are the justifications for choosing a CNN model for this task:

- Input: The datatype (hyperspectral image stacks) is suitable for a CNN compared to other neural networks.
- Labeling: In our case, the CARS spectrum (input), the NRB shape (input component), and the resonant spectrum (target output) are defined for model

training. The model already knows what the target output should look like, so it backtracks its calculation by changing the filter values (weights) after every training step, so the predicted output is closer to the target vector. This process is referred to as "Backpropagation" in literature.

- Feature extraction and classification: The most important justification for using a CNN is its ability to retrieve information from input with the aid of shared weights among filters and its ability to generalize the extracted information on its own [3].

3.3.1 Methodology

The goal of this study is to explore how well this approach performs at removing NRB from experimentally measured CARS hyperspectroscopy and to determine how best to modify this approach to suit the data obtained from our unique SF-CARS scheme. To do this, two DL models were trained on two different sets of realistic simulated CARS spectra. The first model "Specnet" was trained using the dataset created by using the same procedure seen in [3]. The second model was trained with the modified dataset. Section 3.4.1 will introduce the modification made to the approach and the resulting model is denoted as "Specnet Plus". The neural network and model training was implemented in a Python environment using TensorFlow, a deep learning package. The original Specnet code can be found in a GitHub repository [46]. The modified code can be found in Appendix B.

The following is a chronological summary of the steps taken to accomplish this task, and each step's specifics are provided in the sections that follow.

1. Creating the input datasets (Simulated CARS spectra) using the original approach (Specnet) and then with the updated approach (Specnet Plus).
2. Both datasets were used to train two different DL models.
3. Both models' ability to remove to extract the resonant parts from simulated and experimentally measured data were tested. The finding from both tests is discussed in chapter 4.

3.4 Step 1 (a): Simulating the input data for Specnet

A large set of simulated CARS spectra was used as the input to Specnet model in order to yield a generalizable model that can account for different experimental scenarios. This section discusses the process used to create each CARS spectrum. The process description is a synopsis of the approach reported by Valensise et al [3]. Specnet was configured to accept simulated and measured data with intensity $I \epsilon [0,1]$, and the frequencies were also normalized at $\omega \epsilon [0,1]$.

$$\omega \equiv \frac{\omega - \omega_{max}}{\omega_{max} - \omega_{min}} \tag{3.5}$$

where ω_{max} and ω_{min} , corresponds to the maximum and minimum CARS spectrum frequencies that our system is capable of detecting, as mentioned in Section 2.6. The spectra used in the training dataset were manufactured by randomly sampling 15 CARS resonances and, for each of them, the corresponding amplitude, resonance frequency, and

linewidth. The resonances, amplitude, bandwidth, and vibrational frequency were sampled randomly from these distributions as described using

$$\begin{aligned} N &\sim U(1,15); \\ A_i &\sim U(0.01,1); \\ \Gamma_i &\sim U(0.001, 0.008); \\ \Omega_i &\sim U(0,1). \end{aligned} \tag{3.6-3.9}$$

where $U(x, y)$ denotes the uniform distribution; x is the minimum value and y is the maximum. A_i denotes the amplitude, Γ_i denotes the bandwidth, Ω_i, denotes the vibrational frequency, and N represents the number of CARS resonances. Each resonant spectrum, $\chi_r^{(3)}$, was coded using equation 2.7 and normalized to a maximum value of 1 to avoid model rigidity.

The NRB, $\chi_{nr}^{(3)}$, is represented by the product of two sigmoid functions, σ, whose parameters are also randomly sampled.

$$\chi_{nr}^{(3)}(\omega) = \sigma_1(\omega)\sigma_2(\omega) \tag{3.10}$$

these sigmoid functions are defined as:

$$\sigma_i(\omega) = \frac{1}{1 + \exp\left(-(\omega - c_i)s_i\right)} \tag{3.11}$$

The parameters $\{c_i, s_i, i = 1,2\}$ are randomly sampled to generate a non-uniform background distribution across the spectral range with their amplitudes restricted to values ranging from 0 to 1. Like the random sampling done for $\chi_r^{(3)}(\omega)$, this accounts for various experimental scenarios. Recall that theoretically the NRB is often considered to be constant, but in our experimental setup, where the Stokes beam is a highly structured broadband supercontinuum, the NRB is far from constant.

The input vector, x, is then computed as:

$$x = \frac{|r\chi_r^{(3)}(\omega)+\chi_{nr}^{(3)}(\omega)|^2}{2} + \varepsilon(\omega) \tag{3.12}$$

where, ε, is the normally distributed noise component ($\varepsilon \sim N(0, s)$) and s, is modeled to mimic experimental noise. The factor of 2 normalizes the input vector to the maximum possible value (1), which is obtained for a vibrational resonance, ω_{res}, at these conditions:

$$\max\left(\mathrm{Im}\left(\chi_r^{(3)}(\omega_{res})\right)\right) = 1 \tag{3.14}$$

$$\max\left(\chi_{nr}^{(3)}\right) = 1 \tag{3.15}$$

and

$$\mathrm{Re}\left(\chi_r^{(3)}(\omega_{res})\right) = 0 \tag{3.16}$$

This normalization makes sure that the inputs intensities are distributed throughout the (0,1) range. It is crucial when using the model to process experimental CARS data that all curves are normalized to the global maximum of each batch, ensuring that all other amplitudes are in the (0, 1) range. This ensures the model's ability to extract the resonant part without losing information encoded in the peak relative intensities.

Finally, the target vector, y, (the model output) is expressed as:

$$y = \mathrm{lm}(\chi_r^{(3)}) \tag{3.17}$$

3.4.1 Step 1 (b): Simulating the input dataset for Specnet Plus

Except for the NRB shape used in creating the simulated CARS spectra, the modified dataset used the same simulation parameters as in section 3.4. The use of a double sigmoid function to model the slowly varying frequency response of the NRB is not sufficient for modeling a three-color CARS system [51]. This was the justification for modifying the NRB shape used in the simulation to mimic what we observed experimentally. The non-resonant background spectrum collected from the coverslip used in the experiment demonstrated in ROI 2 of Figure 2.5 is shown in greater detail below in Figure 3.5. A plot of the spectrum's running average was carried out to deduce which shape can be used to mirror our experimentally observed NRB response for simulation. This was also done for

an NRB spectrum collected from a water sample, but only the spectrum obtained from the coverslip in Figure 3.5 is presented in this thesis. For both experiment, two similar spectra with three humped broad peaks of varying amplitudes and widths in both spectra. **These observed spectra shapes were significantly different from the linear line highlighted in Figure 2.2 or the double sigmoid function used in the Specnet approach**.

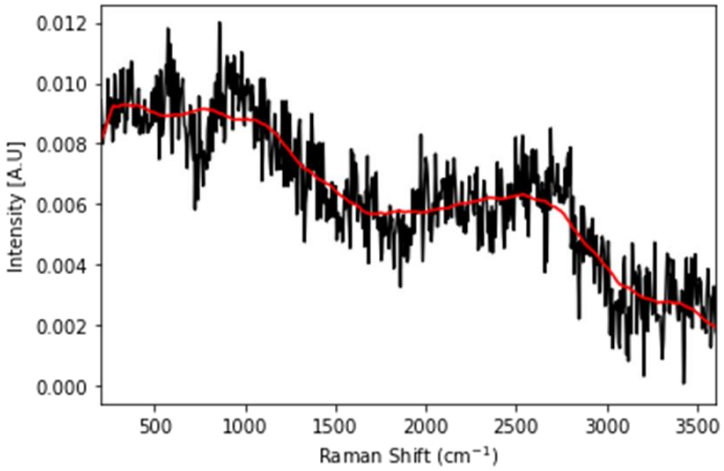

Figure 3.5: CARS spectrum from a glass coverslip as an experimental example of NRB in our system. The spectrum shows a zoomed-in and processed copy of the CARS spectrum from ROI 2 of Figure 2.5. This non-resonant background was collected from the coverslip (black line). A plot of the running average of the spectrum was included to estimate the shape used for simulation (red line). The shape from the running average inspired how the NRB was simulated for the input vector x.

Four gaussian-like peaks with varying widths and amplitudes were simulated to recreate the shape seen in the collected NRB spectra. Four peaks were used to improve the model's generalisation capability. The peak used in each instance were chosen at random to cover

a wide range of experimental conditions. The gaussian peaks amplitudes were sampled from random numbers between 1 and 4,

$$A_{NRB} \sim U(1,4) \quad (3.17\text{-}3.18)$$

where, U, denotes the uniform distribution. For each amplitude, A_{NRB}, the assigned width is sampled from this distribution:

$$\Gamma_{NRB} \sim U(0.1,0.3), \quad (3.17\text{-}3.18)$$

where, Γ_{NRB}, denotes the width for each peak. The NRB shape was added to the simulated CARS spectrum (input x) for each permutation. The dataset with updated NRB spectra was used to train the Specnet Plus model. Chapter 4 will discuss how this modification performed at extracting the resonant spectrum from our experimentally measured CAR spectrum compared to the original approach.

3.5 Model Training and Evaluation

The two datasets created from Step 1 were used to train both Specnet and Specnet Plus models; the datasets consisted of 30000 simulated CARS spectra at 640 spectral points. **The spectral points correspond to the hyperspectral images taken at different**

frequencies. The model's architecture consists of five 1-dimensional CLs with 128, 64, 16, 16, and 16 filters of dimensions 32, 16, 8, 8, and 8, respectively, followed by three FC layers of 32, 16, and 640 neurons (as the output is expected to have the exact dimensions of the input) [3]. Please see Appendix A.1 for a table summarizing the original approach's architecture which is the same for the Specnet Plus model. ReLU was included at the end of each CL and FC layer to add non-linearity to the network. Backpropagation was performed using Adam (a tool for optimizing training performance and pace) with a batch size of 256 examples [3]. The resulting number of trainable parameters was 6×10^6 [3]. Mean squared error (loss function) was computed to quantify the prediction error between the target vector y and the predicted one, $\hat{y}$. To avoid overfitting and reduce the model's sensitivity to noise, L2 weight regularization is utilized on the weights of the first fully connected layer with a weight of 5×10^{-6} [3]. The training required 10-fold cross epochs, each taking about 10 seconds on a GeForce GTX 1080 Ti graphic processing unit (GPU). Figure 3.5 illustrates how backpropagation is done at every training step. The initial predictions, which might be poor, are improved by updating the filters (weights and biases) to make better predictions; each iteration is known as an epoch.

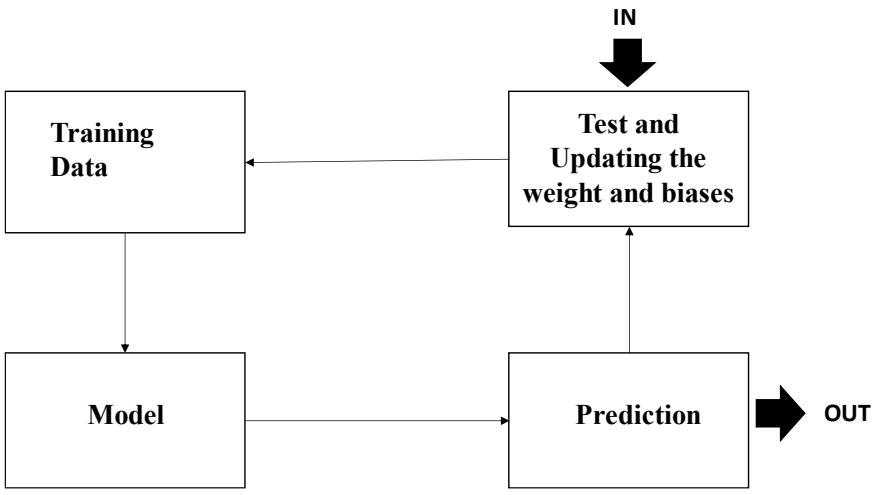

Figure 3.6: A summary of each training step. After an initial assigned weight and bias are used for the first prediction, the training goes through several cycles (or epochs) of updating the weights and biases to predict an output closer to the target vector. The arrow (IN and OUT) indicates the transitions from one cycle to the next.

3.6 Testing both models' prediction performance

The final stage in this approach involves testing both trained models' viability at extracting resonant signals from other simulated CARS spectra and measured SF-CARS hyperspectroscopy. Appendix A.2 provides the code required to execute this test on the measured data. Each pixel in the hyperspectral stack is processed at each spectral data point. The predicted output from each processed pixel is accumulated together to form a completely new hyperspectral stack, termed the "**retrieved hyperspectral stack** or **retrieved stack**". Figure 3.7 show a process flow diagram for testing the model on measured CARS hyperspectral data.

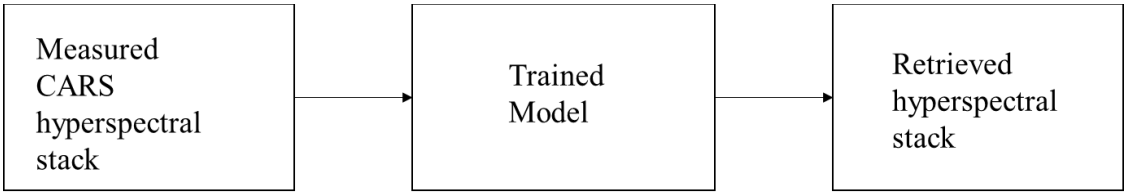

Figure 3.7: A process flow diagram of how the model is used for processing. Both trained models are used to process the measured hyperspectral stack to extract the resonant signals from each pixel. The retrieved hyperspectral stack is created by compiling the processing results from each frame.

Chapter 4

Experimental Results

4.1 Chapter Preface

This chapter reports the ability of both the published and modified Specnet model to extract resonant signals from two randomly simulated spectra and measured SF-CARS hyperspectroscopy on a sample containing potato starch granules and a chitin sample obtained from the exoskeleton of a shellfish.

4.2 Simulated CARS spectra

A preliminary test of each model's ability to extract the resonant spectrum from unknown simulated spectra was conducted. Two extra (random) CARS spectra that weren't used in the model's training were simulated for this test. The two spectra were simulated using the same steps described in section 3.4, which means the resonances were randomly assigned while creating them (i.e., the resonant spectrum for each simulation is known). Mean squared error analysis was employed to provide a quantitative assessment of how each model's prediction output differed from the "known" resonant spectrum (target vector) through the spectral range. The x-axis' frequency scale was normalized between 0 and 1; the Raman shift might be viewed as relative as a result. This translates to 0 cm^{-1} for the first spectral data point and 1 cm^{-1} for the 640th spectral data point.

Figure 4.1 (top panel) shows the prediction results from processing the first CARS spectrum with the Specnet and Specnet Plus, respectively. The first simulation comprised a CARS spectrum with a single resonance at 0.6 cm^{-1}. Although Specnet successfully extracted the resonance peak at 0.6 cm^{-1}, the retrieval of weak spurious signals across the whole spectral range reduced its performance. These spurious signals were retrieved at around 0.3 cm^{-1} and in the spectral region 0.5 to 0.8 cm^{-1}, excluding the resonant frequency. Specnet Plus also retrieved the resonance at 0.6 cm^{-1} and it also retrieved spurious signals at the same spectral regions, but they were significantly weaker in intensity. The mean squared error for Specnet Plus was discovered to be 5 times lower than the value for Specnet.

The second simulated spectrum had six resonances at frequencies around 0.01, 0.48, 0.52, 0.7, and 0.75 cm^{-1}. The strongest peak is located at 0.52 cm^{-1} followed by the peaks at 0.01, 0.48, 0.7, and the resonance at 0.75 cm^{-1} has the lowest intensity. Figure 4.1 (bottom panel) shows the test result from the second CARS spectrum. The resonances were retrieved by Specnet at the same frequencies but differed in intensity and shape in some regions. The peak height at 0.01 cm^{-1} was lower than the resonant spectrum, and the peak shape at 0.75 cm^{-1} looks to still be distorted, indicating that the NRB presence wasn't completely eradicated. Specnet Plus also retrieved a similar resonance as the resonant spectrum. The resonance at 0.01 cm^{-1} had a higher intensity in the Specnet Plus-retrieved resonance spectrum compared to the Specnet-retrieved resonance spectrum, and the resonance at 0.75 cm^{-1} appeared to be less distorted. The mean squared value for Specnet was 2 times greater than that of Specnet Plus. The error analysis results from both tests

indicate that Specnet Plus outperformed Specnet in terms of predicting the target vector (resonant spectrum).

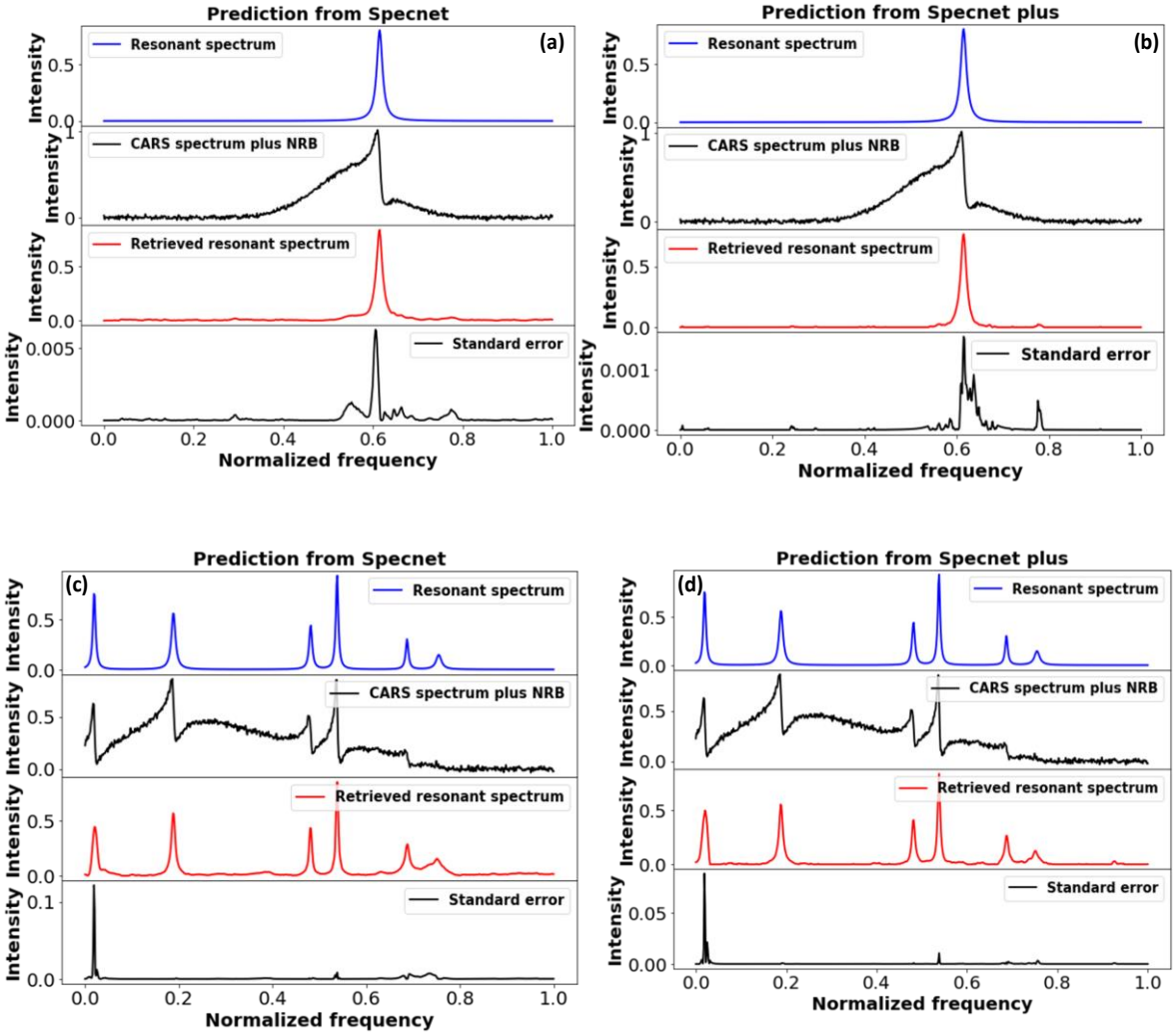

Figure 4.1: The prediction results obtained from using the Specnet and Specnet Plus models on two randomly simulated CARS spectra. Top panel: The predictions result from processing the first CARS spectrum (a and b). Bottom panel: The predictions result from the second test (c and d). Each plot in Figure 3.5 has four subplots that show the resonant spectrum (top), the CARS spectrum with the NRB added (second), the retrieved spectrum

by the model (third), and the squared error (bottom). Blue curves are the simulated "ground truth" and the red curves are the model's attempt at retrieving the true spectrum.

4.3 Experimentally measured CARS data

The ability of each model to extract the Raman spectrum from experimentally recorded CARS hyperspectroscopy is reported in this section. Qualitative analysis is carried out by comparing the retrieved resonances with the vibrational Raman mode assignment in the literature.

4.3.1 Starch

For a variety of reasons, starch was chosen as a sample of interest. The fact that it is the most abundant carbohydrate (a polysaccharide) in many plant cells is one example [47]. Starch is made up of amylose and amylopectin chains and exhibits several Raman resonances including molecular vibrations in the C-H and -OH regions [47]. Because starch is included in many everyday foods, such as potatoes, rice, beans, and others, as well as being simple to prepare, getting a sample is inexpensive and straightforward. To prepare the starch sample, raw potatoes were cut into pieces, steeped in water, stirred, allowed to settle for a short period of time, and then aspirated with a pipette into a fresh vial.

Three 250×250 CARS images of the potato starch sample are shown in Figure 4.2 (top panel). These images were generated by averaging eight frames within a hyperspectral stack centered around 3011 cm^{-1}. The first stack contained the experimentally-measured

raw CARS hyperspectroscopy. The second stack was the retrieved hyperspectral stack that was obtained after the Specnet model was used to process the original stack. The third stack was the processing results from the modified model (Specnet Plus). The pump power and the PCF input were set at 120 mW and 180 mW, respectively. These powers are weak enough to avoid photobleaching and sample degradation. The measured hyperspectral stack was acquired by scanning the delay stage which translates to 467 spectral data points spanning frequencies approximately between 50 cm^{-1} to 3600 cm^{-1}. One of the training parameters (the data points size) was adjusted to fit the measured experimental data (It was changed it from 640 points, as seen in section 3.5, to 467 spectral data points as it is our measured data size). This allows us to retrieve a hyperspectral stack with the same spectral points as the measured stack.

Spectral analysis:

Starch's characteristic Raman resonances at 3200, 2900, 930, and 477 cm^{-1} are associated with molecular vibrations in amylose and amylopectin molecules [48]. The two strongest resonances, at around 477 cm^{-1} and between 2900 and 3100 cm^{-1}, are attributed to the vibrations in the pyranose ring of the glucose and the C-H stretching, respectively. Bands around 860 cm^{-1} and 930 cm^{-1} are assigned to the CH_2 deformations, and the C-O-C vibrational mode [48]. The bands around 3100 cm^{-1} to 3600 cm^{-1} for starch are assigned to the -OH stretching [49].

The measured CARS spectrum derived from the highlighted region of interest (Figure 4.2 bottom panel) shows strong resonances at 488 cm^{-1} and 3011 cm^{-1}, but the non-resonant

contribution severely distorts them. The -OH stretching assignment was seen at 3209 cm^{-1}. It is challenging to locate the resonance at 930 cm^{-1} since it appears to be overpowered by the NRB shape. The CARS spectra obtained from both retrieved hyperspectral stacks, derived from the same region of interest, are also presented in Figure 3.9 bottom panel. Both of the strong resonances, at 488 cm^{-1} and 3011 cm^{-1} were retrieved by both models and thus appear less distorted with a more Lorentzian-like peak compared to the measured CARS spectrum; the Lorentzian-like shape does indicate proper Raman retrieval. Furthermore, obtaining CARS signals below 800 cm^{-1} has been quite difficult historically [50]. The resonance at 3209 cm^{-1} was retrieved by Specnet Plus but Specnet failed to retrieve it. Specnet did predict a new peak at 3313 cm^{-1}, which had a higher peak height than the anticipated resonance at 3209 cm^{-1}. The new peak retrieved by Specnet is most likely a misinterpretation that occurred during processing, according to the resonance assignments found in literature. The resonance at around 930 cm^{-1} was not retrieved by either model, suggesting the vibrational mode was never excited. This non-excitation may be a result of insufficient Stokes light was provided at that frequency. The resonances retrieved between 1000 cm^{-1} and 3000 cm^{-1} were neglected during analysis because they are not included in the characteristic resonance used for identifying starch [48]. The CARS spectrum from the Specnet Plus model appears to be less noisy than that from Specnet, suggesting that the applied modification resulted in a better reduction of observed NRB and image denoising.

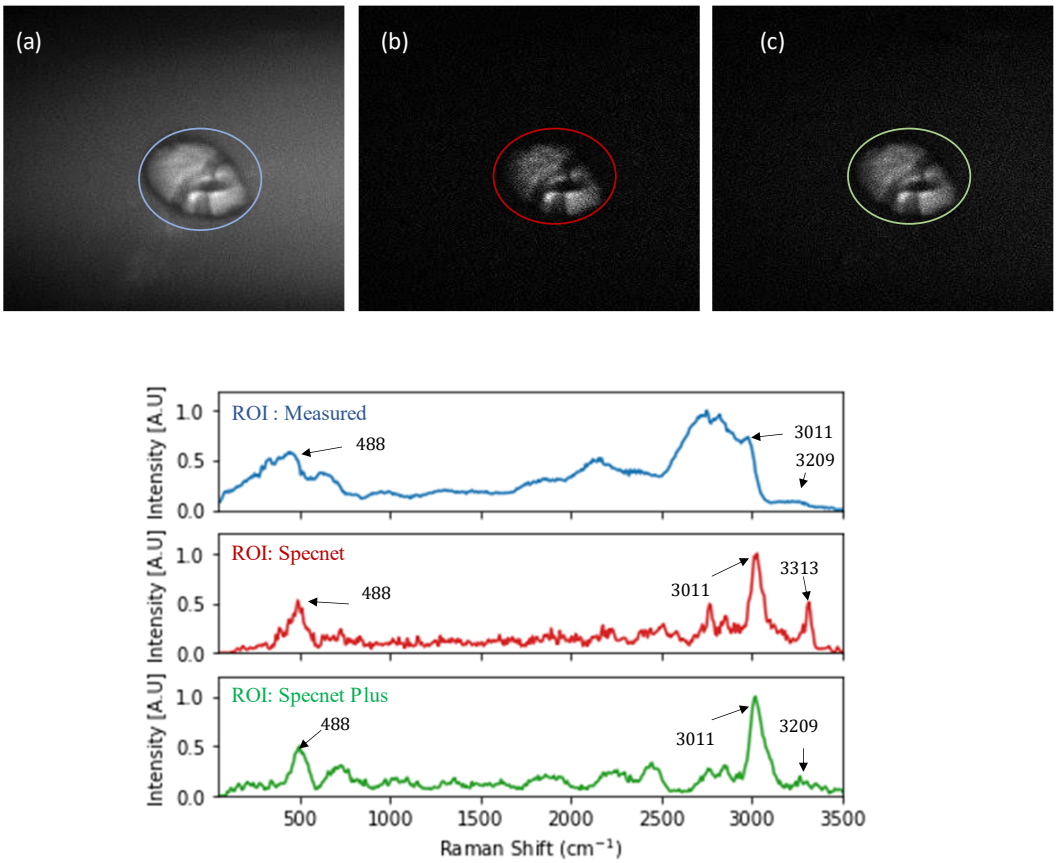

Figure 4.2: SF-CARS hyperspectroscopy of a potato starch grain and the retrieved hyperspectral data from Specnet and Specnet Plus. Top panel: The three CARS images presented were obtained from averaging the same eight frames centered around 3011 cm^{-1} in each hyperspectral stack. (a) Image obtained from the measured CARS hyperspectral stack, (b) Specnet processed image, and (c) Specnet Plus-processed image. Bottom panel: the CARS spectra obtained from the highlighted region of interest in (a-c). The measured CARS spectrum (blue line), the Specnet processed CARS spectrum (red line), and the Specnet Plus processed CARS spectrum (green line).

Image analysis:

Since every pixel at every spectral data point in the original stack is processed by the model by updating them with its prediction, it could be a good idea to establish a metric that compares not only the spectrum but also how the approach influenced imaging. Structural similarity index metric (SSIM) was employed to perform this image analysis; an objective measure that quantifies the overall difference in pixel magnitudes between a processed image and its original. The three CARS images presented in Figure 4.2 (top panel) was used for this test (these images were taken at a resonant frequency). The test resulted in a similarity score of 25.8% between the measured and the Specnet processed image, whereas the test reported a 34.3% similarity between the measured and the Specnet Plus-processed image. Images obtained at a different frequency deemed off-resonance were also used in this test to see how it performs with primarily NRB signals present. The second set of images was collected by averaging eight frames around 3310 cm^{-1}, which was the frequency where the unknown peak was retrieved by Specnet during spectral analysis (Figure 4.3). This region is deep into the O-H vibrational region and is thus reflective of water content or of NRB. The result was a similarity score of 9.98% between the measured and Specnet-processed images. A 7.63% similarity was obtained from testing the measured with the Specnet Plus-processed image. These test results do suggest NRB reduction during processing because of the low similarity score observed from both retrieved images compared to the original, but **they do not give a clear answer as to which model performed better.**

For images collected at resonance, the similarity score is expected to be closer to the original image, whereas images taken off-resonance should have a low similarity score.

The results from both models supported this hypothesis, which raises the possibility that the Specnet Plus may have performed better but these similarity scores by themselves cannot be used to determine which prediction was more accurate. The best way to use this measure to determine which model had a better prediction performance is to integrate some level of supervision by specifying which frame contains predominately the resonant signals and the ones that have the non-resonant signals.

The contrast disparity seen in the two retrieved images in Figure 4.3 demonstrates how both models differed in removing NRB. The image obtained from the measured stack shows a higher contrast in the regions around the starch granule which is predominantly water. The Specnet-processed image shows little starch contrast whereas the retrieved Specnet Plus stack showed no contrast in both regions. The image analysis didn't yield a definite answer to which model performed better at retrieving the resonant signals, but it was worth noting it did show that this approach reduced our experimentally measured NRB.

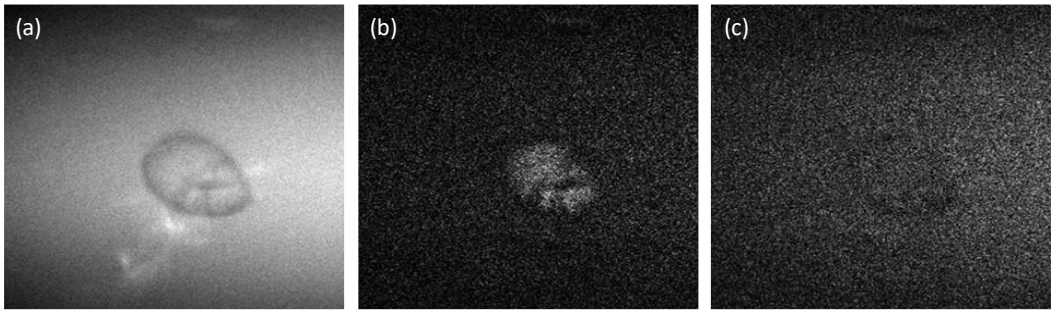

Figure 4.3: SF-CARS images of the potato starch sample gotten at an off-resonance frequency. The three CARS images were obtained by averaging the same eight frames centered around 3313 cm^{-1} in each hyperspectral stack. (a) Image obtained from the measured hyperspectral stack, (b) Specnet processed image, and (c) Specnet Plus processed image.

4.3.2 Chitin

The second most prevalent carbohydrate after starch, chitin is made up of repeating units of N-acetylglucosamine (Nag), the amide derivative of glucose [47]. Chitin can be found in arachnids, insect exoskeletons, shells of crustaceans, and invertebrates (an example of this is shrimp) [47]. Because chitin is a carbohydrate similar to starch but has several other distinct molecular vibrations such as amide stretching, it was chosen as a sample for imaging. The chitin sample was obtained from the exoskeleton of a shrimp. Figures 4.4 (top panel) shows three 250×250 CARS images generated by averaging twelve frames centered at around 2670 cm^{-1} with each hyperspectral stack; the measured hyperspectral stack, the Specnet retrieved stack, and the Specnet Plus retrieved stack. The pump input was set to 75 mW and the PCF input was set to 180 mW; these powers were weak enough to avoid photobleaching and sample degradation. Imaging was done by scanning from one

point on the delay stage to another which translated to 567 spectral data points collected at different frequencies spanning 0 to 3700 cm^{-1}.

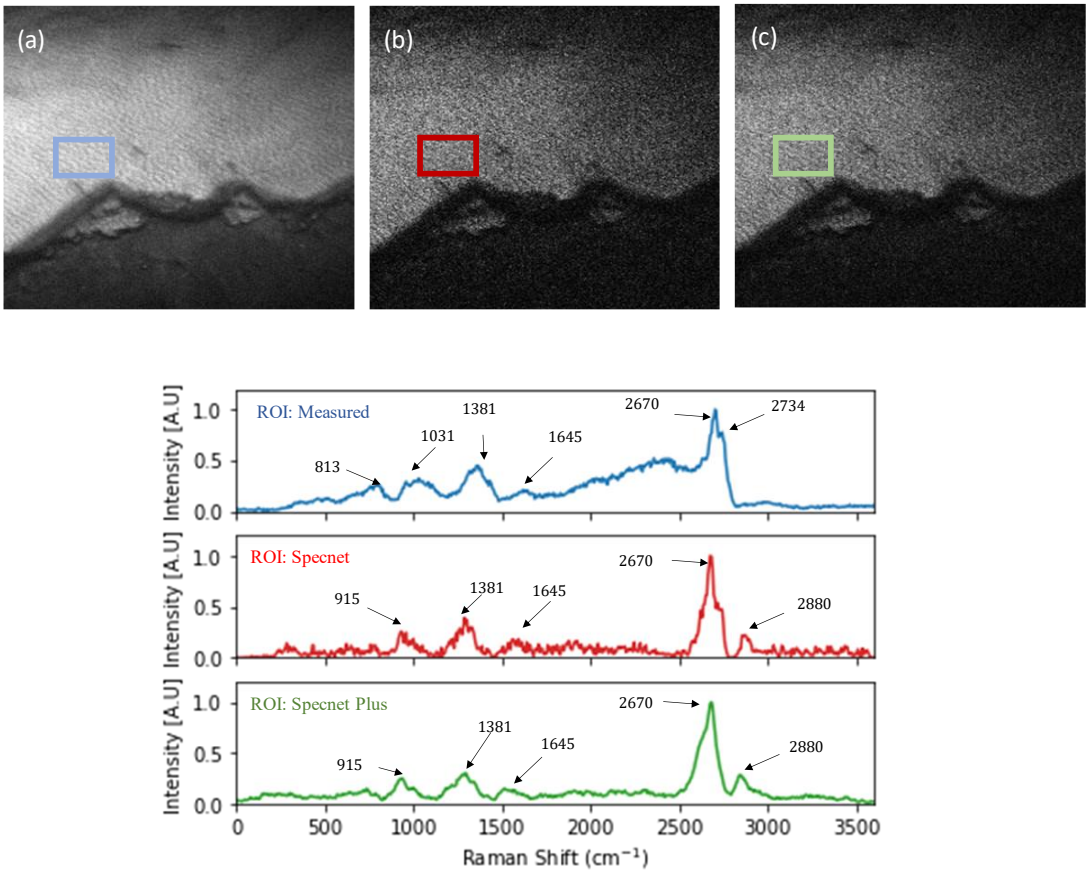

Figure 4.4: SF-CARS hyperspectroscopy of a chitin sample and the retrieved hyperspectral data from Specnet and Specnet Plus. Top panel: The three CARS images presented were obtained from averaging the same twelve frames centered around 2670 cm^{-1} in each hyperspectral stack. (a) Image obtained from the measured hyperspectral stack, (b) Specnet processed image, and (c) Specnet Plus processed image. Bottom panel: the CARS spectra obtained from the highlighted region of interest in (a-c). The measured CARS spectrum (blue line), the Specnet processed CARS spectrum (red line), and the Specnet Plus processed CARS spectrum (green line).

Spectral analysis:

Table 3.1 summarizes the Raman resonance assignments for Chitin found in the spectrum obtained from the region of interest of Figure 3.11.

Frequency Range (cm^{-1})	**Assignment**
800-970	C-O-C stretching
1000-1200	C-O Stretching/ Alicyclic chain stretching
1380-1470	Methylene bending/ C-O-H in-plane bending
1500-1680	Amide I and C=O stretching
2700-3000	C-H stretching

Table 3.1: Summary of detected Raman resonance for Chitin. [51], [52]

Figure 4.4 (bottom panel) shows the experimentally measured CARS spectrum from the highlighted region of interest. The resonance detected at 2670 cm^{-1} is the C-H stretching vibration of methyl and methylene groups of the pyranoid ring [51]; the resonance detected at 2734 cm^{-1} can be attributed to the CH_2 stretching [51], [52]; and the amide and C=O stretching detected at 1645 cm^{-1}. Methylene bending and C-O-H in-plane bending assignment are found at 1381 cm^{-1}; the C-O stretching or alicyclic chain is found at 1031 cm^{-1}; and the C-O-C stretching detected at 813 cm^{-1} [51]. Similar resonances are visible in the CARS spectra from the same region of in the Specnet and Specnet processed stacks (Figure 3.11 bottom panel) at 1381 cm^{-1}, 1645 cm^{-1}, and 2670 cm^{-1}. The strong denoising of the large distortions in the silent region between 2000 cm^{-1} to 2500 cm^{-1} suggests that NRB was reduced by both models. The previously reported C-O-C and C-O contributions, which were found at both 813 cm^{-1} and 1031 cm^{-1}, appear to have been

interpreted as NRB during processing. Both models extract a peak at 915 cm^{-1}, indicating that C-O-C contribution is, in fact, present at this wavenumber. Looking at the two CARS spectra from the Specnet and Specnet Plus stacks, the band at 2734 cm^{-1} was retrieved at 2880 cm^{-1}, illustrating both resonances are present at a later frequency but were reshaped by NRB. Both models predicted similar vibrational peaks, but the modified model appears better suited for our SF-CARS setup, with a less noisy retrieved spectrum compared with the Specnet model.

Image analysis:

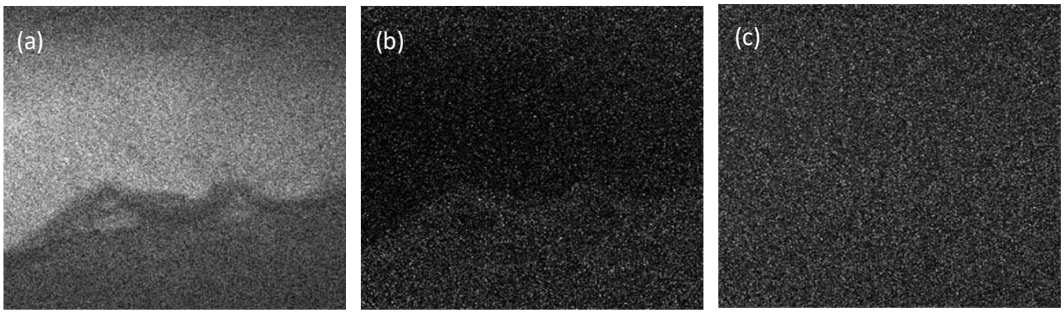

Figure 4.5: SF-CARS images of the chitin sample at an off-resonance frequency. The three CARS images were obtained by averaging the same twelve frames centered around 2450 cm^{-1} in each hyperspectral stack. (a) Raw CARS image, (b) Specnet processed image, and (c) Specnet Plus processed image. The lack of contrast in (c) suggests that Specnet plus model worked as expected.

SSIM was also employed to compare images of the chitin sample that were taken both on and off resonance. The chitin CARS images presented in Figure 4.4 (top panel) taken at a resonant frequency (obtained by averaging twelve frames centered at around 2670 cm^{-1})

was used for the initial test. The comparison between the original image at resonance and the Specnet processed image at the same frequency yielded a similarity score of 33.0%, while comparing the original image with the Specnet Plus processed image was 39.9%. Three CARS images shown in Figure 4.5 were generated by averaging twelve frames centered around 2450 cm^{-1} (at a frequency off-resonance) in a hyperspectral stack. The same test on the images obtained off-resonance yielded a similarity score of -6.50% and -10.10% for the Specnet and Specnet Plus models, respectively. A negative SSIM score indicates that processed images differ significantly from the originally measured one. This result also agreed with the test hypothesis mentioned in section 4.3.1 (image analysis) indicating Specnet Plus may have performed better. The contrast difference between the three images in Figure 4.5 also shows that the non-resonant signals were removed more effectively using the Specnet Plus. The Specnet Plus image showed no contrast at all while the Specnet image still showed a dark contrast at the region contain the chitin sample.

Chapter 5

Summary, Conclusion, and Future Work

This work explores a deep learning approach, the Specnet framework, proposed by others in [3] to remove non-resonant background from measured B-CARS spectra to determine:

- How practical is this approach at retrieving the vibrationally resonant signals and reducing the presence of NRB observed in our SF-CARS setup?
- How best to integrate it into our spectral analysis workflow?

The Specnet model was built as a convolutional neural network consisting of 7 hidden layers. The model was trained on a large dataset of realistic simulated CARS spectra to achieve high generalization capabilities when interpreting and extracting unwanted NRB signals. The NRB shape used to simulate the input dataset (the simulated CARS spectra) was changed to mirror the NRB response observed experimentally to better integrate the proposed approach into our spectral analysis workflow; the resulting model from this modification was termed Specnet Plus. Both models (Specnet and Specnet Plus) were used to retrieve resonant spectra from two randomly simulated spectra and measured CARS hyperspectral stacks from our setup. Since the original resonant spectrum of the simulated spectra is known due to how they were made, an error analysis was done to provide a quantitative assessment of each model's performance. When compared to Specnet, the retrieved resonant spectra from Specnet Plus were more consistent with the original resonant spectra and had a smaller average mean squared error value for both cases. This

preliminary test confirmed that modification improves performance; both qualitatively and quantitatively.

The presence of reduced distortions in the spectra shape in the retrieved spectra suggests that the proposed Specnet framework eradicates the non-resonant signals from our experimentally measured SF-CARS spectra. Also, the approach was able to retrieve most of the identified Raman resonances for both test samples. However, the retrieved Specnet Plus spectra appears to be less noisy than that of Specnet suggesting the modification resulted in better denoising (i.e., performed better at removing the non-resonant background from each CARS spectrum). The structural similarity index metric (SSIM) results also indicated the presence of lesser noise in the retrieved stacks from Specnet Plus. The results from the image analysis indicate updated NRB model agreed with the test hypothesis, but some level of supervision is needed to deduce which model actually performed better. The contrast difference observed in the processed images demonstrates how the two models' interpretation of the resonant signals varied. Although the model with the best performance could not be determined with certainty using the contrast difference and image similarity score, it is still important to note how the modification changed how the measured data was processed. The results from the preliminary test and the appearance of a less noisy retrieved resonant spectra from the measured CARS hyperspectroscopy suggest performance was enhanced by modifying the approach to emulate spectra obtained from our CARS platform, indicating that future work should indeed expand on this methodology.

A recommendation made by others in [53] is to use measured spectra recorded from our CARS platform for model training for better prediction, but this would be cumbersome

work and would require a large volume of measured data. Furthermore, it is not strictly possible to obtain a ground truth Raman spectrum for a given recorded CARS spectrum [53]. As highlighted in [15], the relationship between Raman and CARS should be used with caution, as the non-linear susceptibility probing mechanism is different from the Raman process. Any algorithm used to retrieve a CARS spectrum would necessarily be an approximation, which would compound errors in the retrieval network [53]. Tailoring the simulation parameters such as amplitudes, widths, and frequencies to emulate measured spectra across the spectral ranges will be vital in reducing these approximation errors. The results show that when the simulated NRB shape is changed to match the measured spectra, performance improves. This supports the suggestion that in future work, the input data be tailored to mimic the measured spectra.

Other deep learning approaches, as seen in the work done by Wang *et al.* [53] and Abdolgadher *et al.* [6], which are both based on unsupervised learning should also be explored to determine if they are better suited for our setup. Any improvements to this work will only positively affect CARS imaging and spectral analysis.

Bibliography

[1] S. Li, Y. Li, R. Yi, L. Liu, and J. Qu, “Coherent Anti-Stokes Raman Scattering Microscopy and Its Applications,” *Front Phys*, vol. 8, p. 515, Dec. 2020, doi: 10.3389/FPHY.2020.598420/BIBTEX.

[2] M. D. Duncan, J. Reintjes, and T. J. Manuccia, “Scanning coherent anti-Stokes Raman microscope,” *Opt Lett*, vol. 7, no. 8, p. 350, Aug. 1982, doi: 10.1364/OL.7.000350.

[3] C. M. Valensise, A. Giuseppi, F. Vernuccio, A. de La Cadena, G. Cerullo, and D. Polli, “Removing non-resonant background from CARS spectra via deep learning,” *APL Photonics*, vol. 5, no. 6, Jun. 2020, doi: 10.1063/5.0007821.

[4] Y. Liu, Y. J. Lee, and M. T. Cicerone, “Broadband CARS spectral phase retrieval using a time-domain Kramers–Kronig transform,” *Opt Lett*, vol. 34, no. 9, p. 1363, May 2009, doi: 10.1364/ol.34.001363.

[5] M. T. Cicerone, K. A. Aamer, Y. J. Lee, and E. Vartiainen, “Maximum entropy and time-domain Kramers-Kronig phase retrieval approaches are functionally equivalent for CARS microspectroscopy,” *Journal of Raman Spectroscopy*, vol. 43, no. 5, pp. 637–643, May 2012, doi: 10.1002/JRS.3169.

[6] P. Abdolghader *et al.*, “Unsupervised Hyperspectral Stimulated Raman Microscopy Image Enhancement: Denoising and Segmentation via One-Shot Deep Learning NRC-uOttawa Joint Centre for Extreme Photonics, Ottawa ON K1N 6N5 Canada”.

[7] A. Downes and A. Elfick, “Raman Spectroscopy and Related Techniques in Biomedicine,” *Sensors 2010, Vol. 10, Pages 1871-1889*, vol. 10, no. 3, pp. 1871–1889, Mar. 2010, doi: 10.3390/S100301871.

[8] W. L. Peticolas, “Application of Raman spectroscopy to biological macromolecules,” *Biochimie*, vol. 57, no. 4, pp. 417–428, Jun. 1975, doi: 10.1016/S0300-9084(75)80328-2.

[9] L. P. Choo-Smith *et al.*, “Medical applications of Raman spectroscopy: from proof of principle to clinical implementation,” *Biopolymers*, vol. 67, no. 1, pp. 1–9, 2002, doi: 10.1002/BIP.10064.

[10] R. R. Jones, D. C. Hooper, L. Zhang, D. Wolverson, and V. K. Valev, “Raman Techniques: Fundamentals and Frontiers,” *Nanoscale Research Letters 2019 14:1*, vol. 14, no. 1, pp. 1–34, Jul. 2019, doi: 10.1186/S11671-019-3039-2.

[11] A. Cantarero, “Raman Scattering Applied to Materials Science,” *Procedia Materials Science*, vol. 9, pp. 113–122, Jan. 2015, doi: 10.1016/J.MSPRO.2015.04.014.

[12] “Raman Spectroscopy.” https://integratedoptics.com/Raman-Spectroscopy (accessed Oct. 02, 2022).

[13] R. Smith, K. L. Wright, and L. Ashton, “Raman spectroscopy: an evolving technique for live cell studies,” *Analyst*, vol. 141, no. 12, pp. 3590–3600, Jun. 2016, doi: 10.1039/C6AN00152A.

[14] A. L. Fussell, A. Isomäki, and C. J. Strachan, “Nonlinear Optical Imaging-Introduction and Pharmaceutical Applications”.

[15] B. R. Masters and P. T. C. So, “Handbook of Nonlinear Optical Microscopy,” *PNAS*, 2008.

[16] Y. Wang, C.-Y. Lin, A. Nikolaenko, V. Raghunathan, and E. O. Potma, “Four-wave mixing microscopy of nanostructures,” 2011, doi: 10.1364/AOP.3.000001.

[17] R. Mostany, A. Miquelajauregui, M. Shtrahman, and C. Portera-Cailliau, “Two-photon excitation microscopy and its applications in neuroscience,” *Methods Mol Biol*, vol. 1251, pp. 25–42, 2015, doi: 10.1007/978-1-4939-2080-8_2.

[18] M. Rubart, “Two-photon microscopy of cells and tissue,” *Circ Res*, vol. 95, no. 12, pp. 1154–1166, Dec. 2004, doi: 10.1161/01.RES.0000150593.30324.42.

[19] R. Vargas, A. Mosavi, and L. Ruiz, “DEEP LEARNING: A REVIEW,” 2017.

[20] “An introduction to deep learning - IBM Developer.” https://developer.ibm.com/articles/an-introduction-to-deep-learning/ (accessed Jun. 14, 2022).

[21] L. Alzubaidi *et al.*, “Review of deep learning: concepts, CNN architectures, challenges, applications, future directions,” *Journal of Big Data 2021 8:1*, vol. 8, no. 1, pp. 1–74, Mar. 2021, doi: 10.1186/S40537-021-00444-8.

[22] “Supervised vs. Unsupervised Learning: What’s the Difference? | IBM.” https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning (accessed Oct. 06, 2022).

[23] K. Gurney and N. York, “An introduction to neural networks,” 1997.

[24] “What are Neural Networks? | IBM.” https://www.ibm.com/cloud/learn/neural-networks (accessed Jun. 16, 2022).

[25] “ANN vs CNN vs RNN | Types of Neural Networks.” https://www.analyticsvidhya.com/blog/2020/02/cnn-vs-rnn-vs-mlp-analyzing-3-types-of-neural-networks-in-deep-learning/ (accessed Jun. 16, 2022).

[26] R. W. Terhune, P. D. Maker, and C. M. Savage, “Measurements of Nonlinear Light Scattering,” *Phys Rev Lett*, vol. 14, no. 17, p. 681, Apr. 1965, doi: 10.1103/PhysRevLett.14.681.

[27] R. F. Begley, A. B. Harvey, and R. L. Byer, “Coherent anti-Stokes Raman spectroscopy,” *Appl Phys Lett*, vol. 25, no. 7, p. 387, Oct. 2003, doi: 10.1063/1.1655519.

[28] A. Zumbusch, G. R. Holtom, and X. S. Xie, “Three-Dimensional Vibrational Imaging by Coherent Anti-Stokes Raman Scattering,” *Phys Rev Lett*, vol. 82, no. 20, p. 4142, May 1999, doi: 10.1103/PhysRevLett.82.4142.

[29] H. Huang *et al.*, “Coherent anti-Stokes Raman scattering and spontaneous Raman spectroscopy and microscopy of microalgae with nitrogen depletion,” *Biomedical Optics Express, Vol. 3, Issue 11, pp. 2896-2906*, vol. 3, no. 11, pp. 2896–2906, Nov. 2012, doi: 10.1364/BOE.3.002896.

[30] R. W. Boyd, *Nonlinear Optics*. Elsevier Inc., 2008. doi: 10.1201/9781420004694.ch5.

[31] M. Müller and A. Zumbusch, “Coherent anti-Stokes Raman Scattering Microscopy,” *ChemPhysChem*, vol. 8, no. 15, pp. 2156–2170, Oct. 2007, doi: 10.1002/CPHC.200700202.

[32] Jeremy Porquez, “ADVANCED BROADBAND CARS MICROSCOPY BASED ON A SUPERCONTINUUM-GENERATING PHOTONIC CRYSTAL FIBER,” PHD dissertation , Trent University, Peterborough, 2019.

[33] C. H. * Camp, Y. J. Lee, and M. T. Cicerone, “Quantitative, Comparable Coherent Anti-Stokes Raman Scattering (CARS) Spectroscopy: Correcting Errors in Phase Retrieval,” 2015.

[34] L. G. Rodriguez, S. J. Lockett, and G. R. Holtom, “Coherent anti-stokes Raman scattering microscopy: A biological review,” *Cytometry Part A*, vol. 69A, no. 8, pp. 779–791, Aug. 2006, doi: 10.1002/CYTO.A.20299.

[35] I. W. Schie *et al.*, “Simultaneous forward and epi-CARS microscopy with a single detector by time-correlated single photon counting,” *Optics Express, Vol. 16, Issue 3, pp. 2168-2175*, vol. 16, no. 3, pp. 2168–2175, Feb. 2008, doi: 10.1364/OE.16.002168.

[36] R. Brakel, V. Mudogo, and F. W. Schneider, “Polarization sensitive resonance CARS spectroscopy,” *J Chem Phys*, vol. 84, no. 5, p. 2451, Aug. 1998, doi: 10.1063/1.450363.

[37] R. Cole, “FREQUENCY-TIME AND POLARIZATION CONSIDERATIONS IN SPECTRAL-FOCUSING-BASED CARS MICROSCOPY,” Trent University, Peterborough, 2021.

[38] C. Zhang and J. X. Cheng, “Perspective: Coherent Raman scattering microscopy, the future is bright,” *APL Photonics*, vol. 3, no. 9, p. 090901, Jul. 2018, doi: 10.1063/1.5040101.

[39] C. L. Evans, E. O. Potma, M. Puoris'haag, D. Côté, C. P. Lin, and X. S. Xie, "Chemical imaging of tissue in vivo with video-rate coherent anti-Stokes Raman scattering microscopy," *Proc Natl Acad Sci U S A*, vol. 102, no. 46, pp. 16807–16812, Nov. 2005, doi: 10.1073/PNAS.0508282102/SUPPL_FILE/08282MOVIE1.MP4.

[40] S. H. Parekh, Y. J. Lee, K. A. Aamer, and M. T. Cicerone, "Label-Free Cellular Imaging by Broadband Coherent Anti-Stokes Raman Scattering Microscopy," *Biophys J*, vol. 99, no. 8, p. 2695, Oct. 2010, doi: 10.1016/J.BPJ.2010.08.009.

[41] J. G. Porquez and A. D. Slepkov, "Application of spectral-focusing-CARS microscopy to pharmaceutical sample analysis," *AIP Adv*, vol. 8, no. 9, Sep. 2018, doi: 10.1063/1.5027273.

[42] M. de Veij, P. Vandenabeele, T. de Beer, J. P. Remon, and L. Moens, "Reference database of Raman spectra of pharmaceutical excipients," *Journal of Raman Spectroscopy*, vol. 40, no. 3, pp. 297–307, Mar. 2009, doi: 10.1002/JRS.2125.

[43] J. Pierna, O. Abbas, P. Dardenne, and V. Baeten, "Discrimination of Corsican honey by FT-Raman spectroscopy and chemometrics," *Biotechnologie, Agronomir, Societe et Environnement*, 2011.

[44] R. Yamashita, M. Nishio, R. Kinh, G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology", doi: 10.1007/s13244-018-0639-9.

[45] K. O'shea and R. Nash, "An Introduction to Convolutional Neural Networks".

[46] "GitHub - Valensicv/Specnet." https://github.com/Valensicv/SpecNet (accessed Aug. 02, 2022).

[47] L. Kong, C. Lee, S. H. Kim, and G. R. Ziegler, "Characterization of Starch Polymorphic Structures Using Vibrational Sum Frequency Generation Spectroscopy," *J Phys Chem B*, vol. 118, no. 7, 2014, doi: 10.1021/jp411130n.

[48] S. Wang and P. Guo, *Botanical Sources of Starch*. Springer Singapore, 2020. doi: 10.1007/978-981-15-0622-2_2.

[49] T. Oniszczuk *et al.*, "Physical assessment, spectroscopic and chemometric analysis of starch-based foils with selected functional additives," *PLoS One*, vol. 14, no. 2, Feb. 2019, doi: 10.1371/JOURNAL.PONE.0212070.

[50] A. F. Pegoraro, A. D. Slepkov, A. Ridsdale, D. J. Moffatt, and A. Stolow, "Hyperspectral multimodal CARS microscopy in the fingerprint region," *J Biophotonics*, vol. 7, no. 1–2, pp. 49–58, Jan. 2014, doi: 10.1002/JBIO.201200171.

[51] E. Praveen, S. Murugan, and K. Jayakumar, "Investigations on the existence of piezoelectric property of a bio-polymer-chitosan and its application in vibration

sensors," *RSC Adv*, vol. 7, no. 56, pp. 35490–35495, 2017, doi: 10.1039/C7RA04752E.

[52] B. Gieroba *et al.*, "Surface Chemical and Morphological Analysis of Chitosan/1,3-β-d-Glucan Polysaccharide Films Cross-Linked at 90 °C," *Int J Mol Sci*, vol. 23, no. 11, Jun. 2022, doi: 10.3390/IJMS23115953.

[53] Z. Wang, K. O' Dwyer, R. Muddiman, T. Ward, C. H. Camp, and B. M. Hennelly, "VECTOR: Very deep convolutional autoencoders for non-resonant background removal in broadband coherent anti-Stokes Raman scattering," *Journal of Raman Spectroscopy*, vol. 53, no. 6, pp. 1081–1093, Jun. 2022, doi: 10.1002/JRS.6335.

Appendix A

A.1 The proposed Specnet model architecture summary [3].

Layer (Type)	**Output Shape**	**Parameter Number**
Batch Normalization	(None, 640,1)	4
Activation	(None, 640,1)	0
Convolution Layer 1 (Using Conv1D)	(None, 609, 128)	4,224
Convolution Layer 2 (Using Conv1D)	(None, 594, 64)	131,136
Convolution Layer 3 (Using Conv1D)	(None, 587, 16)	8,208
Convolution Layer 4 (Using Conv1D)	(None, 580, 16)	2,064
Convolution Layer 5 (Using Conv1D)	(None, 573, 16)	2,064
Fully Collected Layer 1 (Dense)	(None, 573, 32)	544
Fully Collected Layer 2 (Dense)	(None, 573, 16)	528
Flatten layer	(None, 9168)	0
Dropout Layer	(None, 9168)	0
Fully Collected Layer 3 (Dense)	(None, 640)	5,540,480
Total number of parameters: 6, 016, 932 Number of Trainable Parameter: 6, 016, 930 Number of Non-trainable Parameter: 2		

A.2

This code used processing both measured hyperspectral stack with the Specnet and Specnet Plus model is presented below.

```
import tensorflow as tf
import os, time
import tifffile
import numpy as np
from scipy.interpolate import interp1d
import pathlib
import multiprocessing as mp
import tensorflow as tf
# import keras.backend as K
from keras.models import Sequential
from keras.layers import Dense, Conv1D, Flatten, BatchNormalization, Activation, Dropout
from keras import regularizers

model = tf.keras.models.load_model('model/467_model.h5')
model.summary()
x_values = np.arange(467)
# x_values= np.arange()

def open_tiff(file_path):
    # *This step changes the tiffile pulled numpy array from 2-D to 3-D
    image = tifffile.TiffFile(file_path)
    pages = image.pages
    page_shape_rows, page_shape_cols = image.pages[0].shape
    output_image = np.zeros((len(pages), page_shape_rows, page_shape_cols))
    for i in range(len(pages)):
        output_image[i] = image.pages[i].asarray()
    return output_image
```

```
def chi3(data):
    # get the shape of the unprocessed tiff file
    z, y, x = data.shape

    # create an empty array for the Raman-retrieved tiff file
    # z-axis should be 640 data points, same as with the trained model
    new_data = np.empty((467, y, x))

    # generate arbitrary values to the _x ('wavenumber') component of the raw data
    # to be used for interpolation purposes later
    # Note. The retrieval algorithm doesn't need actual wavenumbers but only needs the pixel / frame number.
    _x = np.linspace(0, 467, z)

    # process each pixel of the image
    # process along the row axis
    for i in range(y):

        # process along the column axis
        for j in range(x):

            print(f"processing row: {j} column {i}", end="\r")

            # extract the raw CARS spectrum at pixel j, i
            _y = data[:, i, j]


            # re-map the spectrum to 640 points to be compatible with the model
            # _x is 640 points
            # _y is whatever number of data points
            f = interp1d(_x, _y)
```

```
            # retrieve the Raman spectrum

              new_data[:, i, j] = model.predict(f(x_values)[np.newaxis, :,
np.newaxis]).flatten()


    return new_data

# process the files in this directory
rootdir = r"C:\Users\damil\OneDrive\Desktop\pharma"

# get time-stamp to measure how long the process will take
print(time.strftime('%Y_%m_%d %H:%M:%S'))
for subdir, dirs, files in os.walk(rootdir):
    for file in files:
        filepath = pathlib.Path(subdir, file)
        print(f"processing {file}")
        data = open_tiff(filepath)

        #obtain new file path for processed file
                                                    new_file            =
pathlib.Path(pathlib.Path(file).parent,"results_pharma_test", file)
        new_file.parent.mkdir(parents=True, exist_ok=True)
        processed = chi3(data)

        # Save tiff_file

        tifffile.imsave(new_file, processed)
        #tifffile.imsave(os.path.join("results", file), processed)

print(time.strftime('%Y_%m_%d %H:%M:%S'))
```

Appendix B Copyrighted Material

The code for training the DL model with the modification is shown in this section (**the bolded parts**). The section also includes a copy of the MIT license for the original code, which permits me to use it as I wish for this project.

Source code for building the modified Model

The Original code for the Specnet model can be found in the GitHub Repository [46].

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import tensorflow as tf
```

```
import keras.backend as K
from keras.models import Model, Sequential
from keras.layers import Dense, Conv1D, Flatten, BatchNormalization, Activation, Dropout
from keras import regularizers
from datetime import datetime


max_features = 15
n_points = 640
nu = np.linspace(0,1,n_points)

def random_chi3():
    """
    generates a random spectrum, without NRB.
    output:
          params =  matrix of parameters. each row corresponds to the [amplitude, resonance, linewidth] of each generated feature (n_lor,3)
    """
    n_lor = np.random.randint(1,max_features) # take a random int between 1 and max_features
     a = np.random.uniform(0.01,1,n_lor)  # take a random number between 0 and 1 with shape n_lor
    w = np.random.uniform(0.001,1,n_lor)  # take a random number between 0 and 1 with shape n_lor
    g = np.random.uniform(0.001,0.008, n_lor) # take a random number between 0.001 and 0.008 with shape n_lor

    params = np.c_[a,w,g]
    return params
random_chi3()

def build_chi3(params):
    """
    buiilds the normalized chi3 complex vector
```

```
    inputs:
        params: (n_lor, 3)
    outputs
        chi3: complex, (n_points, )
    """

           chi3    =    np.sum(params[:,0]/(-nu[:,np.newaxis]+params[:,1]-
1j*params[:,2]),axis = 1)

    return chi3/np.max(np.abs(chi3))

def sigmoid(x,c,b):
    return 1/(1+np.exp(-(x-c)*b))

def generate_nrb():
    number_of_undulations = np.random.randint(1,4)
    a = np.random.random(number_of_undulations)
    a.sort()
    x0 = np.random.random(number_of_undulations)
    sigma = np.random.uniform(0.1,0.3,number_of_undulations)
    sigma.sort()
    print(f"x0 :\t{x0}")
    print(f"a :\t{a}")

    print(sigma)
    _all = np.exp((-(nu-np.c_[x0])**2)/(2*(np.c_[sigma]**2)))
    result = np.sum(_all,axis=0)

    return result/result.max()

# test function
#plt.plot(generate_nrb())

# generate nrb and store in variable, _nrb
_nrb = generate_nrb()
```

```
# plt.plot(_nrb)

def get_spectrum():
    """
    Produces a cars spectrum.
    It outputs the normalized cars and the corresponding imaginary part.
    Outputs
        cars: (n_points,)
        chi3.imag: (n_points,)
    """
    chi3 = build_chi3(random_chi3())*np.random.uniform(0.3,1)
    nrb = generate_nrb()
    # nrb = _nrb
    noise = np.random.randn(n_points)*np.random.uniform(0.01,0.03)
    cars = ((np.abs(chi3+nrb)**2)/2+noise)
    return cars, chi3.imag

cars, chi3 = get_spectrum()
#plt.plot(cars)
#plt.plot(chi3)
#plt.show()

nrbfig=plt.plot(_nrb)
carsfig=plt.plot(cars)
chi3fig=plt.plot(chi3)

plt.show()

## Seperate the plots here

tf.keras.backend.clear_session()
model = Sequential()

model.add(tf.keras.Input(shape=(n_points,1)))
```

```
# Batch normalization
model.add(BatchNormalization(axis=-1,
                             momentum=0.99,
                             epsilon=0.001,
                             center=True,
                             scale=True,
                             beta_initializer='zeros',
                             gamma_initializer='ones',
                             moving_mean_initializer='zeros',
                             moving_variance_initializer='ones',
                             beta_regularizer=None,
                             gamma_regularizer=None,
                             beta_constraint=None,
                             gamma_constraint=None,
                            # input_shape = (n_points, 1) # can be deprecated
by input layer
                             ))

# gives batch normalization activation 'relu'
model.add(Activation('relu'))

# Conv1D is for pattern detection
# there are 128 kinds of filters and kernel size (32) is convolution window
# adds 128 (filters) to the next dimension
# reduces samples by 32 but adds 1 --> 640 - 32 + 1 = 609
model.add(Conv1D(128, activation = 'relu', kernel_size = (32)))

# do 64 filters, 16 window kernel
model.add(Conv1D(64, activation = 'relu', kernel_size = (16)))

# so on...
model.add(Conv1D(16, activation = 'relu', kernel_size = (8)))
model.add(Conv1D(16, activation = 'relu', kernel_size = (8)))
model.add(Conv1D(16, activation = 'relu', kernel_size = (8)))
```

```
# Dense Layers == Fully connected layers
# 32 units, 573 features
model.add(Dense(32, activation = 'relu',
                kernel_regularizer=regularizers.l1_l2(l1 = 0, l2=0.1)))
# 16 units, 573 features
model.add(Dense(16,             activation             =             'relu',
kernel_regularizer=regularizers.l1_l2(l1 = 0, l2=0.1)))

# Flattens out the previous layer
# 573 * 16 = 9168
model.add(Flatten())

# Dropout. With rate 0.25. Randomly sets a value to zero to prevent
overfitting
model.add(Dropout(0.25))

# Return to previous number of data points
model.add(Dense(n_points, activation='relu'))


model.compile(loss='mse',optimizer='Adam',
metrics=['mean_absolute_error','mse','accuracy'])
model.summary()

def generate_batch(size = 10000):
    X = np.empty((size, n_points,1))
    y = np.empty((size,n_points))

    for i in range(size):
        X[i,:,0], y[i,:] = get_spectrum()
    return X, y

# X, y = generate_batch(10)
# X has shape (10, 640, 1)
# y has shape (10, 640)
```

```
X, y = generate_batch(50000)
history = model.fit(X, y,epochs=10, verbose = 1, validation_split=0.25,
batch_size=256)
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()
model.save('model/640_model.h5')
```