TOWARDS CALIBRATION-INVARIANT SPECTROSCOPY USING DEEP LEARNING

APPLICATIONS OF STATISTICAL LEARNING ALGORITHMS IN ELECTRON SPECTROSCOPY

By

MICHAEL CHATZIDAKIS, B. ENG.

A Thesis Submitted to the School of Graduate Studies in Partial Fulfillment of the Requirements for the Degree of Master of Applied Science

McMaster University Master of Applied Science (2018) Hamilton, Ontario
(Department of Materials Science and Engineering)

TITLE: Applications of Statistical Learning Algorithms in Electron Spectroscopy

AUTHOR: Michael Chatzidakis, B.Eng (McMaster University)

SUPERVISOR: Gianluigi A. Botton

PAGES: xii, 78

Lay Abstract

Spectroscopy is the study of the interaction between photons or electrons and a material to determine what that material is made of. One advanced way to make accurate measurements down to the atomic scale is to use high energy electrons in a transmission electron microscope. Using this instrument, a special type of photograph can be taken of the material (a spectrograph or spectrum) which is detailed enough to identify which kinds of atoms are in the material. The spectrographs are very complicated to interpret and the human eye struggles to find patterns in noisy and low resolution data. Depending on which instrument that the spectrographs are taken on, the resulting spectrograph will also change which adds extra difficulty. In this study, advanced algorithms are used to identify which types of atoms can be identified in the noisy signal from the spectrograph regardless of which instrument is used. These algorithms (convolutional neural networks) are also used in self-driving cars for a similar task of identifying objects whereas in this study we use it for identifying atoms.

Abstract

Building on the recent advances in computer vision with convolutional neural networks, we have built SpectralNet, a spectroscopy-optimized convolutional neural network architecture capable of classifying spectra despite large temporal (i.e. translational, chemical, calibration) shifts. Present methods of measuring the local chemical environment of atoms at the nano-scale involve manual feature extraction and dimensionality reduction of the original signal such as: using the peak onset, the ratio of peaks, or the full-width half maximum of peaks. Convolutional neural networks like SpectralNet are able to automatically find parts of the spectra (i.e. features) of the spectra which maximally discriminate between the classes without requiring manual feature extraction. The advantage of such a process is to remove bias and qualitative interpretation in spectroscopy analysis which occurs during manual feature extraction. Because of this automated feature extraction process, this method of spectroscopy analysis is also immune to instrument calibration differences since it performs classification based on the shape of the spectra. Convolutional neural networks are an ideal statistical classifier for spectroscopy data (i.e. time-series data) due to its shared weighting scheme in neural network weights which is ideal for identifying local correlations between adjacent dimensions of the time-series data.

Over 2000 electron energy loss spectra were collected using a scanning transmission electron microscope of three oxidation states of Mn. SpectralNet was trained to learn the differences between them. We prove generalizability by training SpectralNet on electron energy loss spectroscopy data from one instrument, and test it on a variety of reference spectra found in the literature with perfect accuracy. We also test SpectralNet against a wide variety of high noise samples which a trained human spectroscopist would find incomprehensible. We also compare other neural network architectures used in the literature and determine that SpectralNet, a dense-layer free neural network, is immune to calibration differences whereas other styles of network are not.

Acknowledgements

I would first like to thank my Bachelors thesis and Masters thesis supervisor Dr. Gianluigi A. Botton. He encouraged me to pursue my own ideas and to try and develop new methods to solve open problems in the field of electron spectroscopy. Materials Science & Engineering after all is at the intersection of both applied and fundamental science. I find that too often we emphasize on marginal innovations to existing systems with an industrial engineering focus and we rarely try to do “new” science. In the Botton group however we are encouraged to push the envelope to advance our understanding of electron microscopy and I am thrilled that I had some success applying some pretty hot algorithms (convolutional neural networks) to solve open problems found in electron spectroscopy with generalizable applications to any field of spectroscopy.

I wanted to thank my collaborators on my first publication published last year in ACS Nano: Jeff Hoyt, Gianluigi Botton, Sagar Prabhudev, Payman Saidi and Cory Chiang. This was also work that I presented at the University of Notre Dame’s nanotechnology contest NDConnect where I placed 2nd internationally. Jeff Hoyt provided me a lot of guidance on understanding thermodynamics and how to approach scientific writing in addition to being a great mentor. This project, where I was co-supervised by both Jeff Hoyt and Gianluigi Botton was another example where I was strongly encouraged to pursue my own ideas to understand how the universe works.

My lab mates past and present in the Botton group were also vital for many helpful discussions, feedback and banter, Thank you: David Rossouw, Sagar Prabhudev, Alex Pofelski, Isobel Bicket, Edson Bellido, Sam Stambula, Hanshuo Liu, Reza Safari, Alfredo Carranco, Viktor Kapetanovic, Eric Daigle, Amin Hashemi, and Shayan Masouleh. I also wanted to thank both Alex Pofelski and Sagar Prabhudev for taking the time to give me feedback on my writing style of this very thesis. In addition, thank you to the staff at the Canadian Centre for Electron Microscopy for assistance for sample preparation and sample acquisition: Andreas Korinek, Andy Duft, Travis Casagrande, Chris Butcher, and Carmen Andrei.

Contents

List of Figures

List of Abbreviations

ADF	Annular Dark Field
CTEM	Conventional Transmission Electron Microscopy
CNN	Convolutional Neural Network
EELS	Electron Energy Loss Spectroscopy
ELNES	Energy Loss Near Edge Structures
eV	Electron Volt
FEG	Field Emission Gun
FWHM	Full-Width Half-Maximum
GAP	Global Average Pooling
PCA	Principal Component Analysis
ReLU	Rectified Linear Unit
SGD	Stochastic Gradient Descent
STEM	Scanning Transmission Electron Microscopy
SpectralNet	Spectroscopy optimized Neural Network
TEM	Transmission Electron Microscopy
t-SNE	t-distributed Stochastic Neighbor Embedding
XAS	X-ray Absorption Spectroscopy
ZLP	Zero-Loss Peak

1 Introduction

A trained human spectroscopist is able to look at an unknown spectrum, which can be thought of as time-series data, overlay a proposed candidate reference spectrum and determine (qualitatively) if there is a match. The human brain can perform this task with ease despite translational shifts in the spectrum or varying levels of noise between the reference spectrum and acquired spectrum. The rigor of matching unknown spectra to references can be improved using generalized linear models, fitting procedures, or cross-correlation functions but great difficulty arises with these methods in situations with high noise or translational shifts. There is a need to develop new generalizable and automated methods which can remove qualitative interpretation in spectroscopy. Qualitative interpretation in spectroscopy adds bias to the analysis which includes any overlaying of reference spectra, manual peak shifting, and manual feature selection such as using the full-width half-maximum of peaks, or the intensity ratio between peaks. In addition to these issues, many present methods of quantification/identification require reference standards from the same instrument or from instruments with the same calibration which severely limits the amount of data available to a human spectroscopist. In this study, we apply advances in *statistical learning algorithms* (also called machine learning, or narrow artificial intelligence) to better identify characteristics of a spectrum.

In the following chapter, Chapter 2, we will describe how *domain experts* (spectroscopists trained in spectroscopy) have used *feature engineering* to develop useful predictor variables (or *features*) which can be used to differentiate between spectra collected from materials that have different oxidation states. Such features include metrics like the width of peaks, ratio of peaks and distance between peaks. Feature engineering is the manual process of a domain expert performing dimensionality reduction by using domain knowledge to isolate key pieces of information from the original data. This is in contrast to using statistical learning algorithms which are able to find features automatically by aggregating statistics over large datasets. These algorithms, as we will learn in Chapter 3, learn to find features automatically by viewing many examples of spectra and deciding which pieces of the spectra are the most useful for differentiating between signals. A description of currently accepted methods will be given in the preceding sections of this chapter. Both manual feature extraction processes using human engineered bench marks as well as qualitative interpretation in spectroscopy whereby a trained human spectroscopist performs classification will both be outlined.

The type of spectroscopy that will be explored in this study is called electron energy loss spectroscopy (or EELS). This method can only be performed in a transmission electron microscope which is a very sensitive instrument used for imaging incredibly small objects (smaller than the nanometer). In the main text of this thesis in Chapter 4, a method will be outlined using neural networks to classify bonding information contained in an EELS spectra. The data collection, model training and model testing will be outlined and a rigorous analysis is presented pertaining to if these tools are useful in electron spectroscopy.

2 Fundamentals of Core-Loss Spectroscopy

2.1 Transmission Electron Microscopy

Transmission electron microscopy (TEM) is an invaluable high-resolution characterization tool for a materials scientist to understand the microstructure and properties of a material. In a traditional light microscope (commonly referred to as "optical microscope"), due to limitations of the diffraction limit, the wavelength of the photon will ultimately limit the resolution of the microscope. At a given wavelength of light, the optical microscope will only be able to discern a feature in the field of view that is roughly half of the size of the wavelength of visible light. This presents a problem for the characterization of materials considering visible light is in the range of hundreds of nanometers and many phenomena that are crucial to understand when characterizing materials may be well below this size.

The diffraction limit is circumvented in a TEM by using electrons instead of photons since the wavelength of the electron can be modulated using the de Broglie relationship below,

$$\lambda = \frac{h}{\sqrt{2mqV}}$$

Equation 1

Given some large electron accelerating voltage V, the wavelength of the resulting electron can be observed to be very small. For a 20 keV electron source, the resulting wavelength is on the order of 0.01 nm which is an enormous increase in resolution when compared to visible light. In this case, however, the resolution is no longer limited by the physics but by the instrument itself, due to intrinsic aberration of the electron optics of the microscope. The resolution, in the best cases today is approximately 20-50 x the de Broglie wavelength of the electron, at 300 keV about 0.5-0.6 Å at best.

A TEM consists of an electron gun (electron source), followed by a series of electromagnetic lenses which shape and bend the beam of electrons until they intersect the sample and pass through (transmit through). This is in contrast to a scanning electron microscope which generates secondary electrons and primary electron "bouncing off" the sample and these are collected by various detectors. Typically after the electron beam transmits through the sample in a TEM, there are detectors after the sample which can measure the electron beam to collect information regarding the interaction between the electron beam and the sample.

There are many different types of electron sources. Two popular styles of electron guns are field emission guns (FEG) and thermionic guns. Thermionic sources like LaB_6 typically have higher probe currents than FEGs but operate at substantially higher temperatures since they are required to heat the filament to emit the electrons.

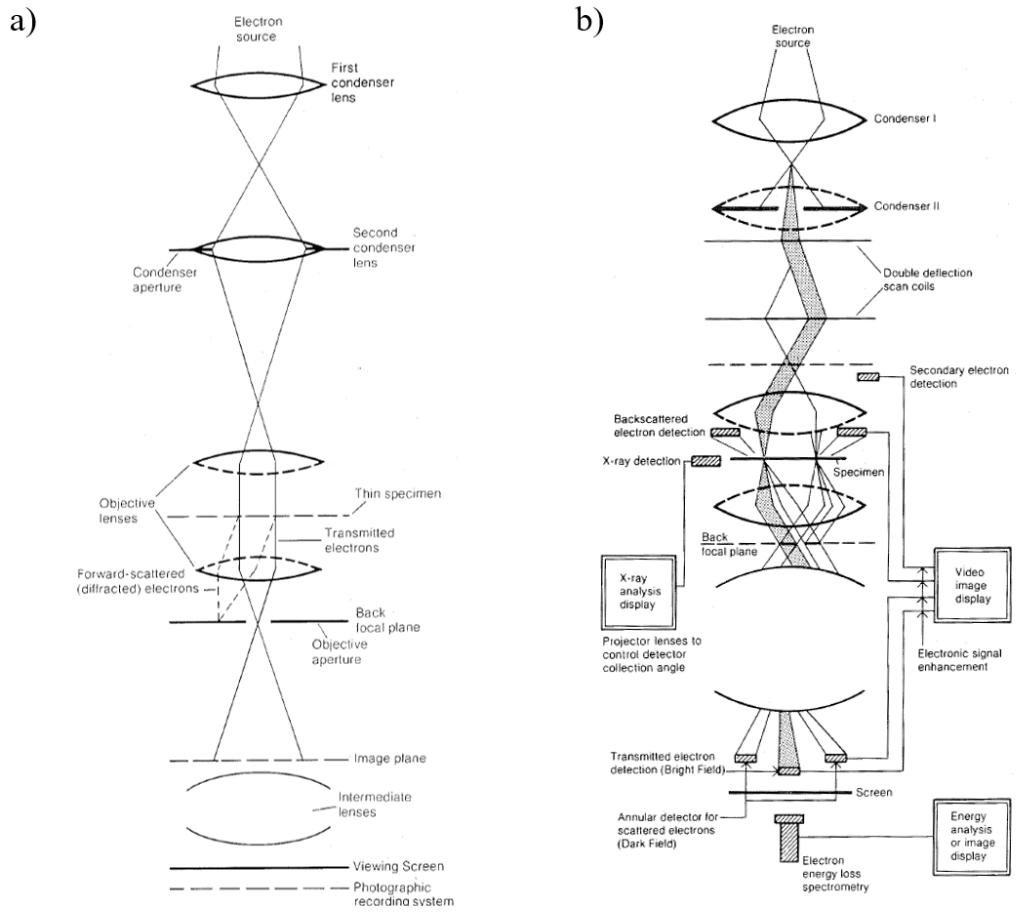

a)
Electron source
First condenser lens
Second condenser lens
Condenser aperture
Objective lenses
Thin specimen
Transmitted electrons
Forward-scattered (diffracted) electrons
Back focal plane
Objective aperture
Image plane
Intermediate lenses
Viewing Screen
Photographic recording system
b)
Electron source
Condenser I
Condenser II
Double deflection scan coils
Secondary electron detection
Backscattered electron detection
X-ray detection
Specimen
Back focal plane
X-ray analysis display
Video image display
Projector lenses to control detector collection angle
Electronic signal enhancement
Transmitted electron detection (Bright Field)
Screen
Annular detector for scattered electrons (Dark Field)
Electron energy loss spectrometry
Energy analysis or image display

Figure 2.1. Schematic of a conventional transmission electron microscope (a) and a scanning transmission electron microscope (b) showing how the electron beam travels through each lens. Images scanned from *Practical Analytical Electron Microscopy in Materials Science* (Williams, 1984).

After a beam of electrons is created using the electron source, this beam is manipulated using a series of electromagnetic lenses. The electron gun itself is a point-source which means it will spread the electrons radially from the tip of the gun. This can be understood in a similar manner as a consumer tungsten filament light bulb radially filling a room with light. The electrons first needs to be shaped and redirected using a series of condenser lenses to produce a concentrated beam. The spread of the electron beams just after generation of the source is shown schematically at the top of Figure 2.1. An aperture (the condenser aperture) is located just after the condenser lenses and is used to control the amount of electrons moving forward. In practice this is one method to control the illumination during acquisition.

The channeled electron beam after leaving the condenser system will either hit the sample as a broad parallel stream of electrons (conventional TEM) or it will converge onto a point on locations of the sample (scanning TEM, or STEM). The differences between these two techniques are shown schematically in Figure 2.1.

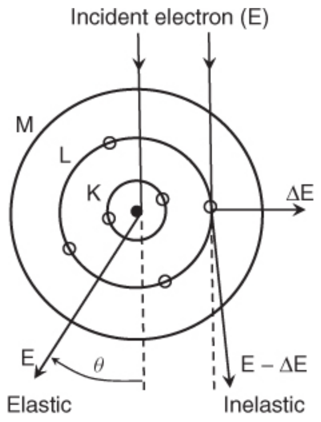


Figure 2.2. Schematic of how an incident electron may scatter elastically or inelastically when striking core-level electrons of the atom. Image scanned from *Low Voltage Electron Microscopy Principles and Applications* (Bell, 2013).

The interaction between the electron beam and the sample can occur in one of three ways. One possibility is that the electron beam may not interact with the sample at all through the samples thickness. The second possibility is that the beam scatters elastically with the sample. The last possibility is that the beam inelastically scatters with the sample. A diagram of inelastic and elastic scattering is shown in Figure 2.2 (Bell, 2013). In the elastic case, the incident electron (i.e. the electron beam) will be deflected by the atomic nucleus at some range of angles dictated by the cross-section function. In the inelastic case, there will also be a deflection, but in addition to the scattering there will also be an energy transfer. The incident electron will impart some kinetic energy (momentum) onto an inner shell electron in the atom thus promoting it to a higher energy level or ionizing (ejecting) it outright. This inelastic scattering will be described in more detail in the following section on *Electron Energy Loss Spectroscopy (EELS)*.

After the electron beam has passed through the sample, the beam passes through the objective, intermediate and projection lenses. After passing through the objective lens, where the first intermediate image is formed, the diffraction pattern in reciprocal space of the sample is created on the back focal plane. The intermediate and projection lenses then magnify and correct the image before the image is passed to either the viewing screen or a camera for recording the image. The electron beam can also be passed to other instrumentation such as a spectrometer for EELS.

2.2 Electron Energy Loss Spectroscopy (EELS)

Electron energy loss spectroscopy is an analytical electron spectroscopy technique capable of providing information regarding a materials electronic structure. This is performed by using information collected from the inelastic scattering of the electron beam in a transmission electron microscope as was discussed earlier in the previous section. In these events where inelastic scattering occurs between the sample and the electron beam, the collected beam will have a small change in its original energy due to the interaction between the beam and the electrons in sample. This is shown schematically in Figure 2.2.

The magnitude of energy loss reflects different phenomena occurring in the interaction. If no energy- loss occurs (or just about zero loss occurs at some standard deviation), this interaction is not from inelastic scattering and is from complete transmittance through the sample. This component of the spectrum is termed the zero-loss peak (ZLP). The ZLP is the majority of the collected signal for very thin samples. The ZLP is also the only fixed reference point and calibration is typically performed using the ZLP as will be discussed in detail in the *Microscope Calibration* section later. Typically it is the width of the ZLP at its full-width half-maximum (FWHM) intensity that is a metric for energy resolution. The reason for this is that all EELS edges will be convolved with the ZLP, so if the spread of the ZLP is large, we can expect a blurring of the ionization edges we are trying to observe in proportion to the size of the ZLP. The ZLP can be seen in Figure 2.3.

In the low-loss region of the electron energy loss spectrum, which is typically under 50 eV, phenomena such as plasmons, intraband or interband transitions can be detected. Plasmons are the collective oscillations of electrons in the conduction band and are useful in probing optical properties of materials. No low-loss investigations were performed in the present study.

The core-loss region of the EELS spectrum (>50 eV) contains characteristic information related to the sample's electronic structure and all present work in this study involves core-loss EELS. As an incident electron strikes a core level electron of an atom in the sample, kinetic energy will be imparted onto the atom. The energy may be sufficient to promote an electron from the valence band to the conduction band or to ionize a valence band electron completely and eject the core electron from the sample. In the case of ionization, the incident electron beam will lose energy in proportion to how much was required to ionize the sample's core-level electrons. This is a precise amount of energy and the amount of energy is different for every electronic configuration thus giving characteristic information related to the electronic properties of the sample. In practice the ionization energy of simple systems like individual atoms or diatomic molecules will be precise. However when taking into account molecular bonding effects in a crystal lattice or complicated molecules, the ionization edge may change slightly between the same species in different compounds. These edge onset fluctuations will be discussed in significant detail in the next section on *Energy Loss Near Edge Structures*.

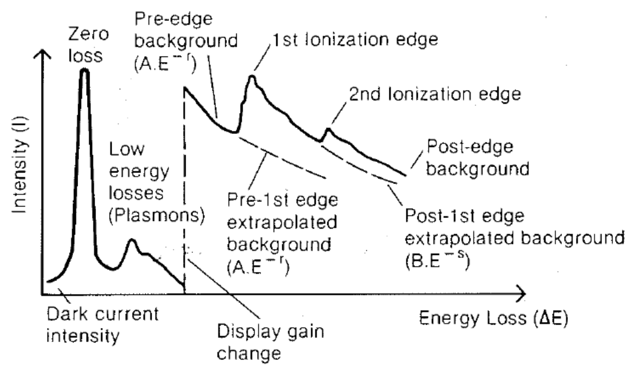

Zero loss
Pre-edge background (A.E^{-r})
1st Ionization edge
2nd Ionization edge
Intensity (I)
Low energy losses (Plasmons)
Post-edge background
Pre-1st edge extrapolated background (A.E^{-r})
Post-1st edge extrapolated background (B.E^{-s})
Dark current intensity
Display gain change
Energy Loss (ΔE)

Figure 2.3. A schematic of the different sections of a typical electron energy-loss spectrum. Image scanned from *Practical Analytical Electron Microscopy in Materials Science* (William, 1984).

In core-loss EELS analysis, it can be observed in the spectrum that a sharp edge exists at the critical ionization threshold and afterwards a drop-off occurs after this point. The sharp edge is the minimum energy required for ionization. It is also possible for additional energy to be imparted as well as ionization in the form of kinetic energy. This is responsible for the tapering off of ionization energy just after the edge. The decrease in intensity after ionization has been shown to be proportional to an exponentially decaying function and typically a power-law is used to least-squares fit this background. The background and extrapolated background are shown in Figure 2.3 before the ionization edge.

2.3 Energy Loss Near Edge Structures (ELNES)

Using the data collected from EELS, it is also possible to extract additional information related to the *unoccupied* bonding states of the sample. During ionization due to the incident electron, the inner-shell electrons of the sample are imparted with enough energy such that they can be leave their ground state orbital. These electrons are ejected despite the nucleus attracting them, and can either be fully ejected from the atom or they may also be promoted to higher unoccupied energy states (Keast, Scott, Brydson, Williams, & Bruley, 2001). Probing the transition of core-level electrons going from the valence to the conduction band shines light on a variety of properties regarding the atoms in the sample ranging from coordination number, the type of bonding and even the oxidation state of the atom. In this study we will exclusively be discussing how the oxidation state is related to the fine structure of EELS signals.

The transition metals on the periodic table of elements are able to accept a wide variety of oxidation states. The oxidation state of an atom is a concept representing how many electrons are missing from a complete electron shell. In many cases, elements may only have one or two possible oxidation states due to the electronegativity of the atom. For example chlorine as Cl^- having a (-1) oxidation state very eagerly wants to donate its one surplus electron to a nearby atom, whereas sodium as Na^+ having a (+1) oxidation state ideally only wants to accept just one electron. Both chlorine and sodium in these examples are very likely just to transfer 1 charge respectively if the charged elements are placed in proximity to each other. This is called an *ionic* bond. That is to say, a total charge transfer will occur between the two elements to fill each of their shells. Purely Ionic bonding is typically not present in transition metals forming compounds with non-transition metal atoms. Many transition metals bond with other transition metal atoms through *metallic* bonds and form bonds with non-transition metals with mixed covalent bonds, meaning electrons (more formally, electron density) are shared between atoms in the compound in the form of a delocalized electron gas (or covalent bonds). This poses a problem in the case of determining oxidation state of transition metals. There may be a discrepancy between what is expected to be the physical charge on the atom (determined by deviance from a full electron shell) and what is observed as the oxidation state because neighboring atoms may pull or push electron density.

Two common characterization methods, used to determine the oxidation state of a transition metal element, are based on probing the unoccupied molecular orbitals as a consequence of core-loss excitation processes: x-ray absorption spectroscopy (XAS) and electron energy loss spectroscopy (EELS). Typically XAS has a higher energy resolution than EELS, but it suffers from low spatial resolution when mapping the location of atoms. Since EELS is generally performed using STEM, it is capable of atomically resolved spatial mapping in addition to high energy resolution (although not as high as XAS) which is invaluable for characterization. Other methods such as Mossbauer spectroscopy or x-ray photoelectron spectroscopy are also very capable of determining oxidation state information but they also suffer from a lack of spatial resolution. Only EELS is used in the present study to tie the fine structure of the unoccupied bonding states of a material to its oxidation state, although in principal XAS would also suffice.

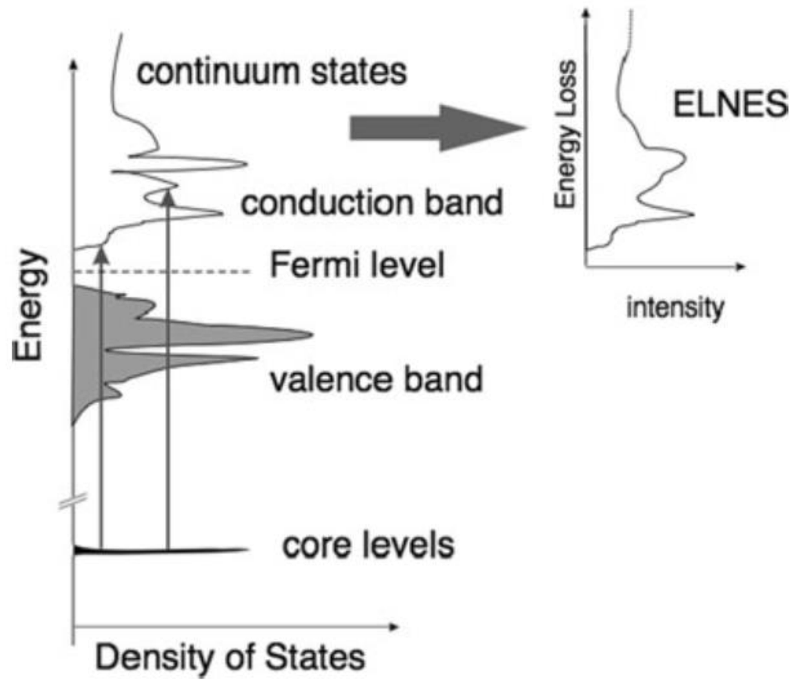


Figure 2.4. A schematic showing how core-level electrons transition into unoccupied states (unshaded) during excitation by the primary electron (Radtke & Botton, 2011).

In EELS, we measure the energy difference between the electron beam before and after it enters the sample. In the inelastic case (refer to the previous section to see the other cases), the electrons in the beam will lose energy in proportion to how much energy was transferred to the sample. The primary electron (i.e. incident electron) loses energy by imparting kinetic energy onto a core-level electron of the sample. If the kinetic energy is very large, then the electron will simply be ejected from the atom and the incident electron will lose energy in proportion to the core-level electrons binding energy plus any additional kinetic energy. However if the kinetic energy transfer is large and there is an unoccupied orbital that is physically accessible to the ionized electron and the transition is allowed, then it is possible to promote that core-level electron to the unoccupied orbital in the conduction band. This is shown schematically in Figure 2.4 above whereby a core level electron can be promoted to the conduction band.

Promoting these core-level electrons to these unoccupied states by the incident electron means that the incident electron will lose at least the energy required for the transition to occur and this is the energy loss that is measured in the spectrum. Determining the energy of these transitions allow the electronic structure of a material to be explored and, in some cases, this will immediately hint what the oxidation state is since a surplus or a deficit of electrons will change the electronic structure of the material. A comprehensive study of how the ionization edge of Mn changes with increasing oxidation state was carried out by Garvie. This systematic variation of the spectra is reproduced below in Figure 2.5. Subtle differences can be observed in the shape, number and width of peaks visible in the spectra for different oxidation states of Mn. In other words, there are systematic "features" in the shape of the peaks existing for the same oxidation state between different minerals. These characteristics shapes gave inspiration for the study using SpectralNet.

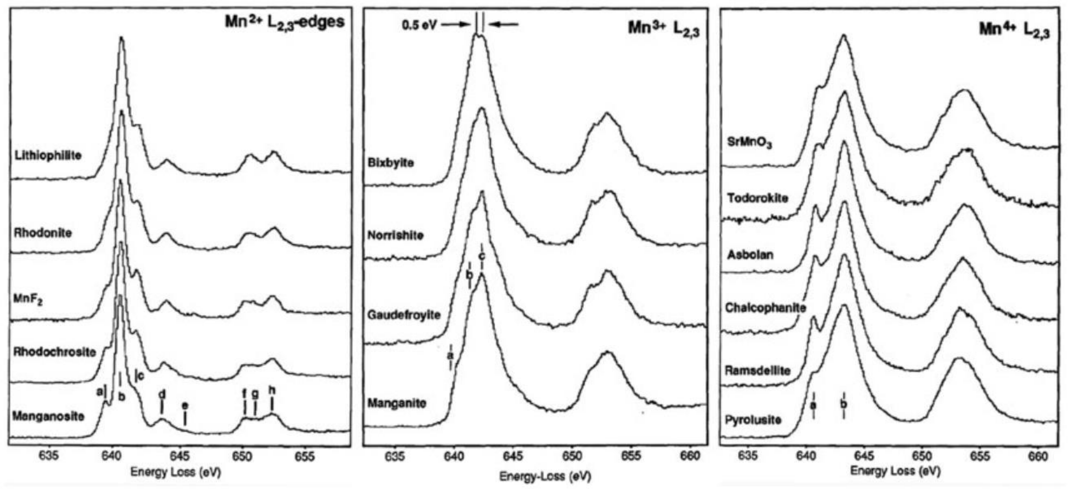


Figure 2.5. Experimental ionization edges ($L_{2,3}$) of Mn taken from Garvie for three oxidation states (Garvie & Craven, 1994).

These core-level to unoccupied bonding state transitions can be observed in the EELS signals as small fluctuations, also termed the *fine structure*, and these are unique to the electronic structure of the atoms in the lattice that we are observing. In the present study we are trying to tie the fine structure of an electronic configuration to its oxidation state. Our method does not involve using domain knowledge from core-loss spectroscopy, instead we rely on artificial neural networks to learn *for us* how the signal changes with respect to oxidation state.

2.4 Microscope Calibration

The calibration of the EELS spectrometer is typically performed by setting the reference point to the zero-loss peak (ZLP). The ZLP signal represents electrons which have passed through the sample with no interaction. Using the ZLP as a reference point, it is the distance between the ZLP and the core-loss edge which we use to index characterizing information regarding the electronic configuration of the observed sample. It is then obvious that if the ZLP were to move as a function of time due to instability of the incident electron energy or spectrometer instabilities, then the calibration would be incorrect since the sample we want to index would have an incorrect reference point. The ZLP does indeed move due to the unstable energy of the electron beam and spectrometer, leading to the inaccuracies in EELS calibration when the ZLP is not acquired at the same time as the ELNES data.

Since a spectrometer has a limited amount of energy channels available (500 to 5000 usually), to achieve a small bin size (i.e. fine energy resolution) then the total available energy window will have to be small. It is then impossible to capture both the ZLP and the core-loss peak within the same energy window since most transition metals are many hundreds of eV (i.e. 650 eV in the case of Mn) away from the ZLP. Achieving good energy resolution per channel (e.g., 0.1 eV/channel) will require many thousands of channels which are out-of-scope for many spectrometers. The commonly accepted practice is to calibrate the spectrometer using the ZLP in one energy window, then move the energy window to the core-loss peaks in question after calibration using the energy drift tube. The beam energy will likely change as a function of time, and constant recalibration would be required to maintain calibration. In addition to this, a new source of systematic error is introduced by moving the energy window since the energy drift tube also requires calibration by the manufacturer.

The fluctuations of the beam will directly translate into fluctuations in the edge onset because of the need to use the ZLP for spectrometer calibration. By increasing the spread of beam energy, the location of the beam center becomes harder to locate. The spread of beam energy are instrument sensitive with the noise profile being distinct for different machines as shown in Figure 2.6 which is reproduced from Potapov and Schryvers (Potapov & Schryvers, 2004). In their study, they show energy variations of the position of the edge onset between 0.05 and 0.5 eV over just a time-span of 40 seconds when comparing three TEMs.

Potapov and Schryvers show that a mono-chromated Tecnai-HR with specialized high-tension circuitry optimized for high energy resolution was measured to have an energy drift of 0.04 eV per minute. Most other TEMs perform significantly worse as shown in Figure 2.6. They calculate that after calibration, in a conventional acquisition of a sample, it would take at least a minute afterwards to switch from a low-loss (ZLP) to a core-loss energy window and to adjust illumination. It would then take multiple minutes to acquire a spectrum over an exposure time and to collect dark current reference images. They estimate that the typical uncertainty in the calibration is expected to be on the order of 0.5 eV or worse by the time an acquisition is made just because of energy drift.

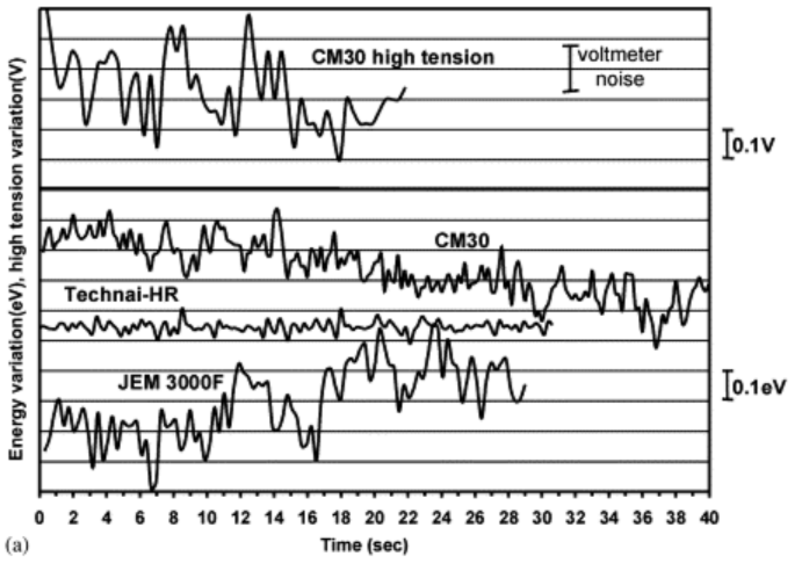


Figure 2.6. The energy variation of the electron beam is compared over time for three different instruments. Reproduced from Potapov and Schryvers (Potapov & Schryvers, 2004).

Having an error of 0.5 eV of the edge onset energy may not seem like a large amount of error, but consider that the difference in edge onset energies as a function of oxidation states is only on the order of only a couple eV. This will be discussed in detail in the following section, but calibration error is one reason why the absolute energy position of the ionization edge is not a reliable method to measure oxidation states in EELS.

2.5 Current methods for valence classification using ELNES

2.5.1 Absolute position of the edge onset

The phenomena of increasing ionization energy with increasing formal oxidation state is termed the *chemical shift*. In practice measuring the absolute edge onset in spectroscopic methods such as XAS is very successful which allows the chemical shift to be the only feature required for oxidation state discrimination. This is not the case in EELS however due to primary energy drift and calibration differences between instruments (Tan, Verbeeck, Abakumov, & Van Tendeloo, 2012). Potapov found that the uncertainty in the calibration of the same instrument between calibration and acquisition can be higher than 0.5 eV even in high-energy resolution microscopes (Potapov & Schryvers, 2004). The difference in calibration between instruments may be substantially higher if the spread of the beam energy differs between the instruments. The difference in edge onset energy between oxidation states is typically only on the order of a 1 to 2 eV (Figure 2.7). When considering the uncertainty in primary energy being 0.5 eV this is a significant amount of error.

Many methods have been developed to try and pin-point the edge onset for one instrument. One method includes using software to automate acquisition by performing multiple calibrations of the zero-loss peak (ZLP) when switching between different core-loss peaks (Tan *et al*., 2012). Acquisition needs to occur quickly after calibration to reduce primary energy drift and Tan *et al* used an automated script to achieve good results. The occurrence of primary energy drift after calibration was discussed in detail in the *Instrument Calibration* section and it is for this reason that the absolute position of the edge onset is not a reliable feature for oxidation state discrimination.

Tan *et al* used special automated software to reduce the time between calibration and acquisition and they had very good results on tying the edge onset energy with the oxidation state as shown in the figure below. Their results however, would not be useful for other microscopy groups with different instruments or calibration due to the very small energy differences between oxidation states (~25% error using uncertainty from Potapov).

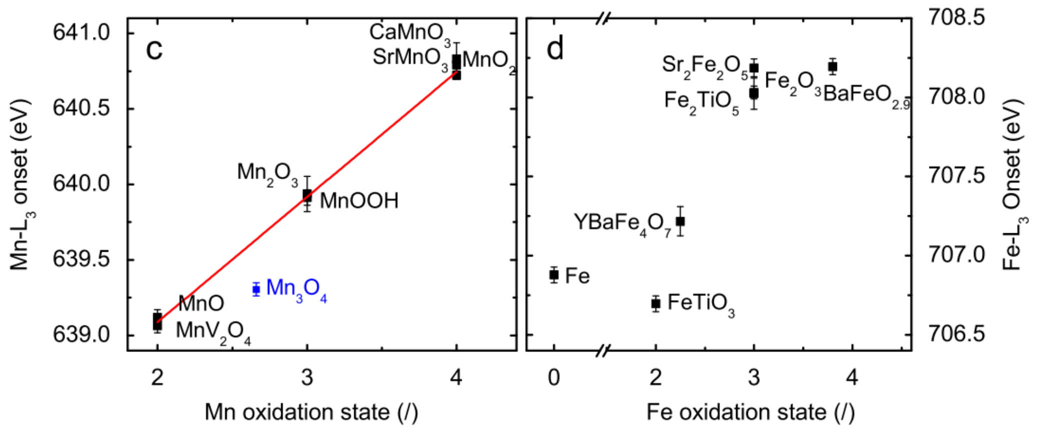


Figure 2.7. The oxidation state of both Mn and Fe are compared as a function of edge onset energy. The difference between oxidation states can be determined by only a 1 or 2 eV edge onset energy difference. Figure adapted from Tan *et al* (Tan *et al*., 2012).

2.5.2 White-line intensity method

The *"white-line"* intensity method also known as the $L_{2,3}$ ratio method, is one of the most popular approaches for oxidation state determination in EELS. In this method, the integrated peak ratio between the L_3 peak and the L_2 peak of a transition metal $L_{2,3}$ edge is calculated which is considered a proxy for probing the $2p_{3/2} \rightarrow 3d$ (corresponding to the L_3 "peak") and $2p_{1/2} \rightarrow 3d$ (for L_2 peak) transitions. This is an example of *manual feature engineering*. Using domain knowledge of core-loss spectroscopy, trained spectroscopists have developed a metric ($L_{2,3}$ ratio) which is used to discriminate between oxidation states. Metrics such as the edge onset energy are considered a discriminating feature.

However, in practice the $L_{2,3}$ ratio method is successful in determining the oxidation states in some compounds containing the transition metals Mn and Fe (Tan *et al*., 2012), but not all compounds or not all oxidation states. This integrated intensity can be a function of sample thickness since the probability of multiple scattering increases with increased thickness. This presents a major challenge for this method and the $L_{2,3}$ ratio is not considered to be an effective or reliable feature representation of the input spectra since it is correlated with other features like the thickness.

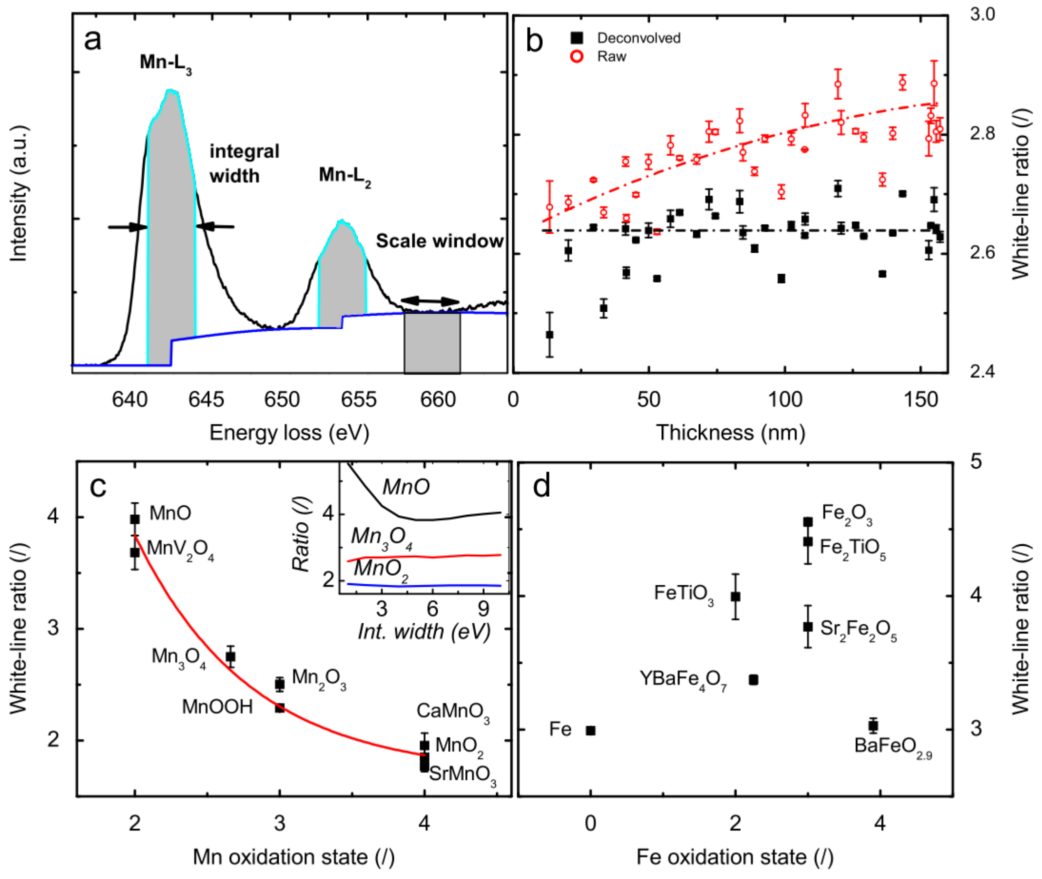


Figure 2.8. a) Example of the integration windows used in the *white-line* intensity method for Mn. b) White line ratio as a function of thickness. Multiple scattering deconvolution is effective in removing multiple scattering effects. c) The correlation between white-line ratio and Mn oxidation state is shown to be non-linear. d) The correlation between white-line ratio and Fe oxidation state is shown to be more difficult to interpret. This figure is reproduced from Tan *et al*.

Figure 2.8 shows that the thickness correlation with the white-line intensity can be reduced using Fourier ratio multiple scattering deconvolution by cleaning the data but the correlation still exists. Without multiple scattering deconvolution, it can be observed from Figure 2.8 that discrimination using the white line intensity would introduce significant error.

2.5.3 Two parameter methods

In the previous two sections we have been introduced to the feature engineering process. Using knowledge of how core-loss spectroscopy works, a domain expert can create a representation of the data in the form of a feature (i.e. white-line ratio, chemical shift) which allows better discrimination between oxidation states. Both the white-line ratio method and the absolute edge position measurements are methods capable of transforming the original high dimensional data into 1 single feature each. In this section we will be combining two manually created features to further differentiate between oxidation states.

Yedra *et al*. created a script in Digital Micrograph, named *Oxide Wizard*, to differentiate the oxidation states for various metal oxides in EELS (Yedra *et al*., 2014). Using domain knowledge they engineer features such as the ratios of peaks, the FWHM of peaks and the distances between oxygen and the metal. By combining the features in pairs they can better discriminate between oxidation states. An example of their two parameter plot is shown below.

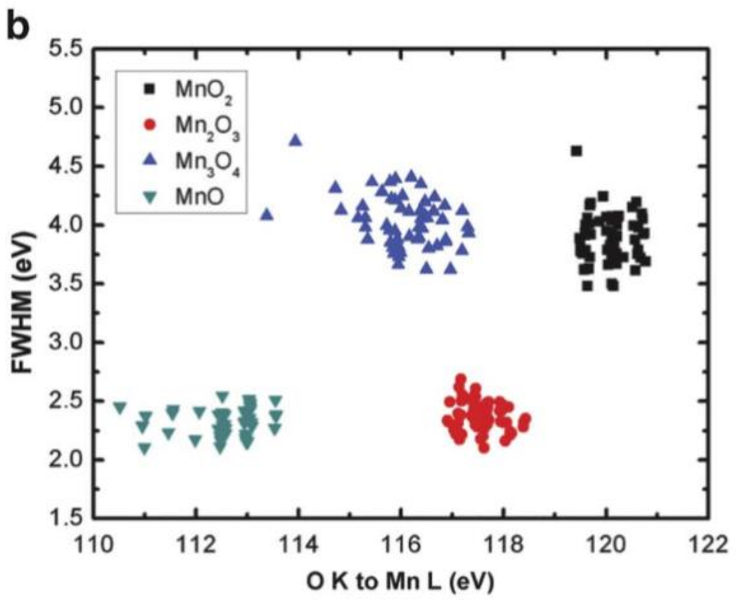


Figure 2.9. Two parameter plot from *Oxide Wizard* used to discriminate between Mn oxides (Yedra *et al*., 2014).

Using the two parameter plot in Figure 2.9, clear separation can be made between four types of Mn oxides. Clear distinction is made between each cluster of points. A simple decision tree can be used to create a set of rules to discriminate between the valence states. An example of such rules can be seen below in a simple decision tree (Figure 2.10).

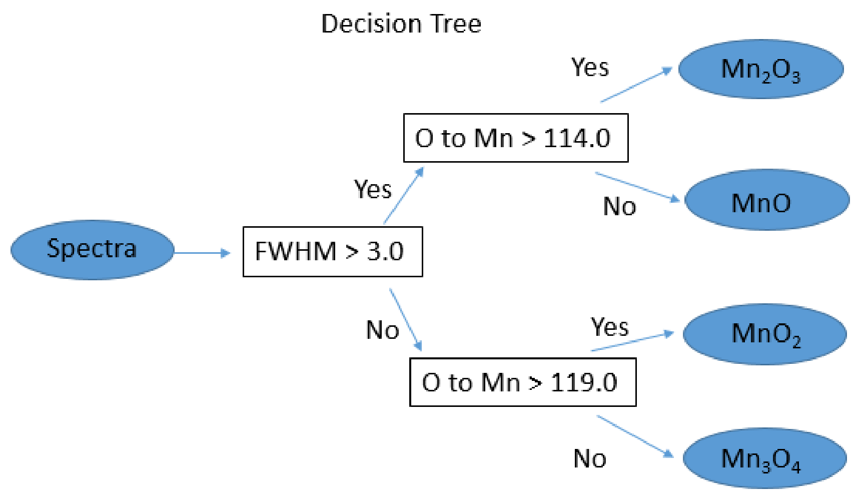


Figure 2.10. Example of a decision tree using the data from **Figure 2.9**

Unfortunately the discriminating features created in Oxide Wizard only apply to oxides. One of the key features they require is the distance between the oxygen and transition metal edges. Because of this, this style of classifier would not work for any other compound such as non-binary systems and metal halides which our classifier hopes to achieve.

2.5.4 ELNES least-squares fitting method

The final method of finding the oxidation state of an unknown compound is to compare the unknown spectra to a reference standard. The comparison can either be qualitative via visual inspection by the researcher or by least-squares fitting the reference standard to the unknown spectra. Both methods are used in the literature and are accepted as accurate methods but both are qualitative analyses (Keast *et al*., 2001; Maigné & Twesten, 2009; Tan *et al*., 2012; Zhang, Livi, Gaillot, Stone, & Veblen, 2010).

Using a least-squares technique involves finding a linear scaling factor a such that a reference standard can approximate the acquired signal. The scaling factor a is found by performing a least-squares (mean-squared error) linear regression by minimizing the error between the reference and the acquired data. One method to minimize error is by using gradient descent. A description of mean-squared error and gradient descent can be found in Chapter 3 on the *Fundamentals of Neural Networks*.

As an example of a typical use-case of least-squares fitting, a grain boundary compound was isolated from a multiple component cathode material in a Li-ion battery. The grain boundary compound has many components and the task was to determine the oxidation state of each element present such that a stoichiometric formula could be determined for the unknown grain boundary compound. A variety of references were least-squares fitted or manually overlaid on top of the unknown spectra as shown in the figure below. The residual between the fitting and the acquired data can be calculated as a description of misfit. However a key problem with this method is making the decision if the reference compound is indeed the grain boundary compound. The titanium edge and the oxygen edges are compared between a strontium titanate signal and the acquired signal. Good match between the Ti signals (the lower energy peaks ~460eV) is observed with a weaker match in the oxygen signal on the right. This information tells us that perhaps the chemical environment of the Ti atoms in both samples are similar, but the chemical environment of oxygen is very different from the reference standard.

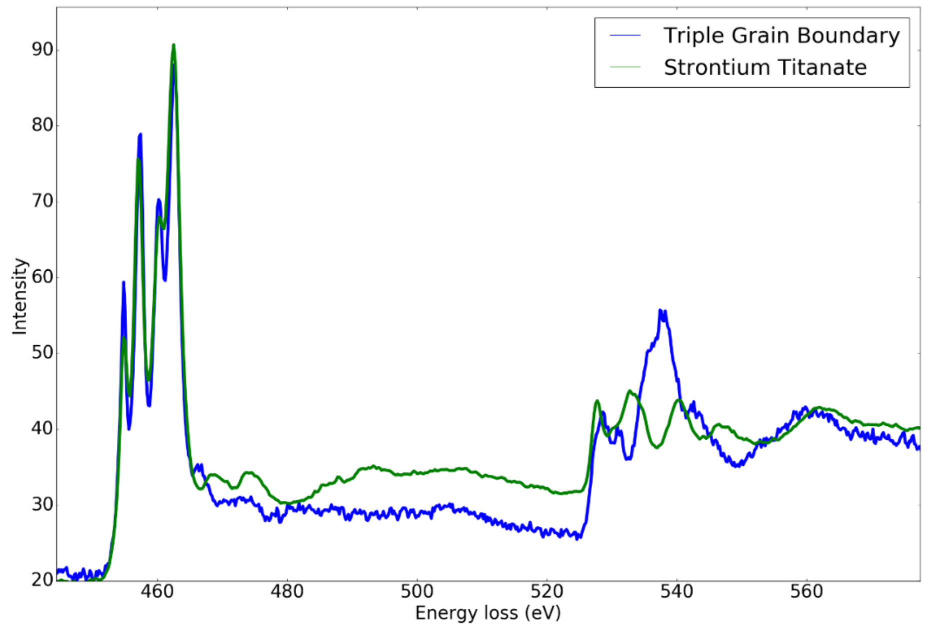


Figure 2.11. An example of using least-squares fitting to overlay a reference standard (strontium titanate) over an unknown acquired signal (triple junction grain boundary segregated compound).

In the example in Figure 2.11, a few qualitative decisions need to be performed by the researcher performing the analysis. The first is deciding which reference standards should be used. If the material is truly unknown as is the case below with non-equilibrium segregated phases, deciding on reference standards is a brute force technique. The second qualitative decision is choosing a boundary of residual error on when we decide if the fit is good or not. The third qualitative decision is by manually performing a chemical/calibration shift between the acquired and reference data. In many cases there will be a small translational shift which will add a significant amount of residual error, so the researcher will be required to shift the data until peak onsets line up. Shifting the data warps the original data and destroys some information. Given these energy scale issues, the goal of using statistical classifiers like SpectralNet is to eliminate all of these sources of error.

3 Fundamentals of Neural Networks

3.1 Overview

This chapter seeks to define some key nomenclature and notations used in neural networks as a prerequisite to understanding the SpectralNet architecture. Neural networks are able to *learn* the relationship between a given input (e.g. an EELS spectrum) and a given output label (e.g., a valence of an element) such that assuming the network has learned through enough examples, it will be able to predict the output label from an unknown input vector. In a similar manner that a Fourier series can model a periodic function, or a Taylor series can model a polynomial, a neural network is capable of modelling any function given enough parameters.

A typical multilayer neural network architecture is schematically shown below in Figure 3.1. The term *architecture* is used here to describe the structure and layout of the units in the graph. The term *multi-layer* (otherwise termed *deep* in *deep* learning) is used to describe the amount of hidden layers in the connected graph network – that is, how many steps and operations exist between input $\boldsymbol{x}$ and output $\boldsymbol{y}$ respectively. The operations performed at each hidden layer will be discussed in detail in the following sections.

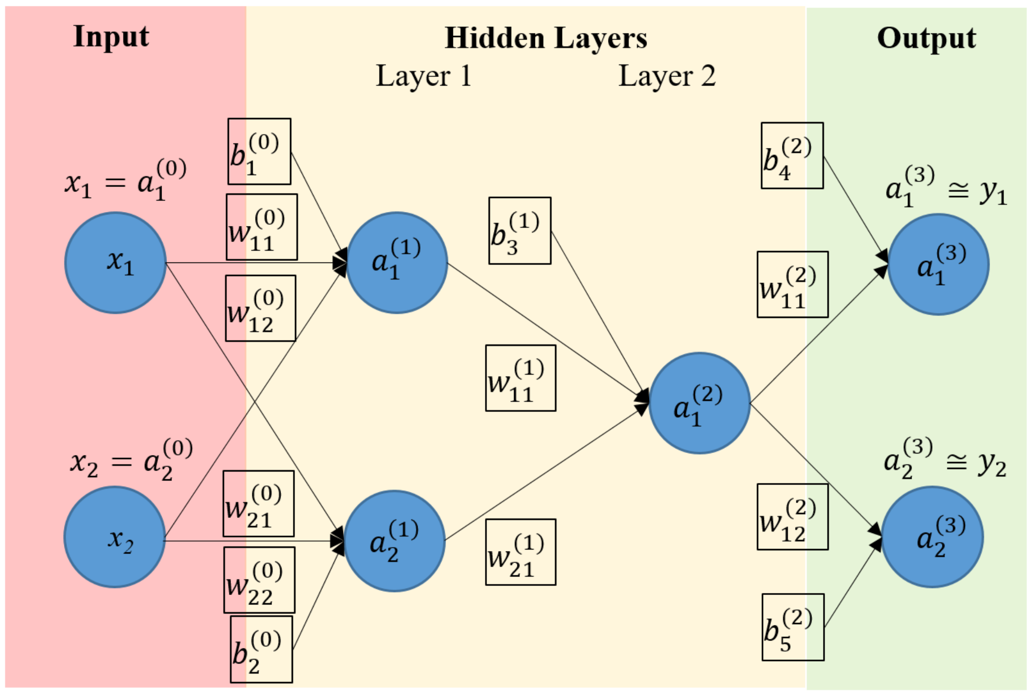


Figure 3.1. An example of a multilayer neural network graph. The hidden layers of the network, which contain free parameters, can be tuned such that the relationship between an input vector (e.g. an EELS spectra) and an output vector (e.g. an atomic valence) can be learned using gradient descent.

In Figure 3.1 directed arrows (termed edges) are drawn between circles (also called neurons, nodes, or vertices) indicating a direction of information flow. There is a direction to operations and the standard notation is that they are read from left to right. The far left neurons are the input neurons and represent an input vector where each neuron holds one real-valued dimension of the vector (e.g., a bin in an EELS spectrum). The far right neurons are the output neurons and represent a vector encoding of the data labels (e.g., the valence of an element). For a classification problem generally the output vector is in the form of a *one-hot encoding* of the categorical variables. Formally, a one-hot encoding of a **k** categorical variable classification would be a sparse vector of length **k** where all but one dimension of this vector is zero.

To connect the input vector to the output vector, it can be observed in Figure 3.1 that there are a series of so-called *hidden layers* between them. Hidden layers consist of many tunable *degrees of freedom* which are used to map the input domain to the output domain. These degrees of freedom are tunable parameters that we try to solve for. The input vector passes through each hidden layer of the network and is transformed using both linear and non-linear operations until it arrives at the output neurons. In a perfectly tuned network, the value of the output neuron will be equal to the label after the input data is propagated forward through the hidden layers.

Directed neural networks like the one shown in Figure 3.1 are typically used in *supervised* classifications settings. Problems that are *supervised* are ones where we have a "question and answer" pair of data. This could be an EELS signal (question) and an oxidation state (answer). This is in contrast to unsupervised learning where we only have a collection of questions, and we are looking to find structure in the data. Such unsupervised methods include techniques that we will discuss in this study such as: k-Means clustering, principal component analysis and t-distributed stochastic neighbor embedding. An example of the type of labelled data for supervised classification that we are using in this study would be the following: for an input spectrum of length 300 (channels), its label would be Mn^{2+}. Mn^{2+} would be required to be represented as a one-hot encoded vector of the form [1,0,0] in the case of a 3 class classification problem ([0,1,0] for Mn^{3+} and [0,0,1] for Mn^{4+}).

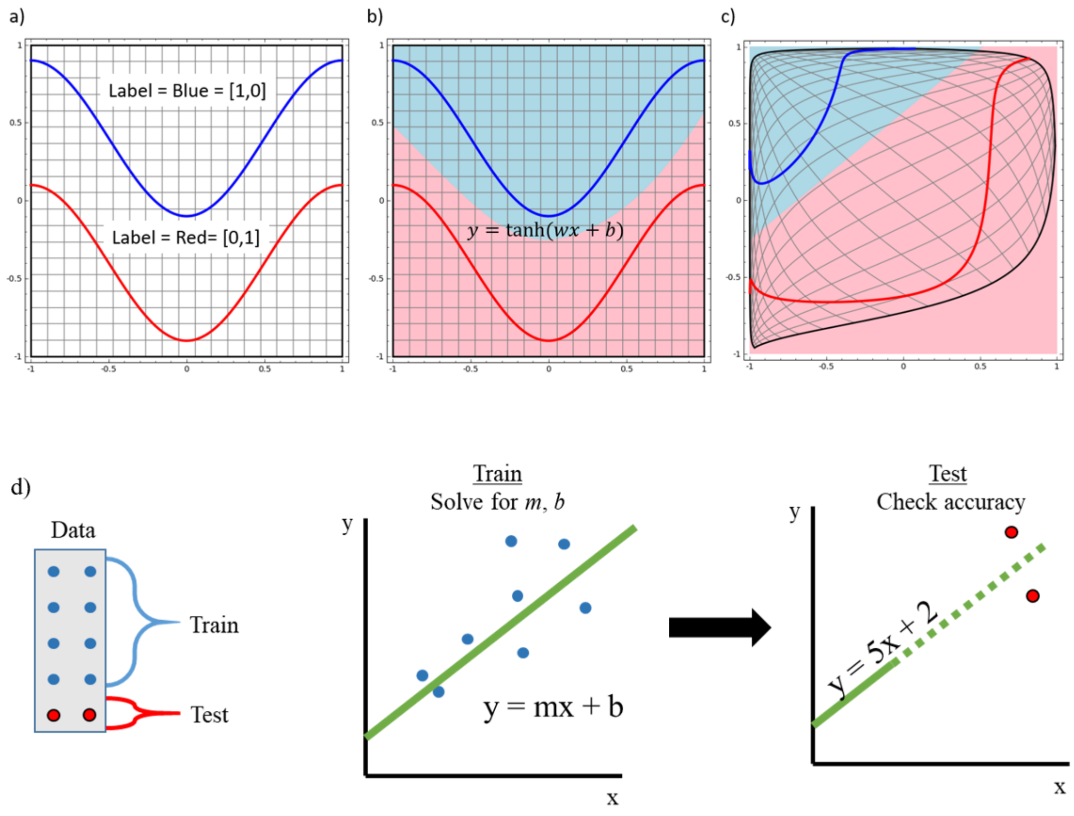


Figure 3.2 a) A 2-dimensional toy classification problem where the points in the blue curve belong to one family and the points in the red curve belong to another family. b) A decision boundary can be drawn on the input domain such that anything above this line will be considered blue, and anything below the line will be considered red. c) By transforming the input domain into one described by the fitted function in (b), a representation can be made of the input domain whereby a simple linear decision boundary can be made. Both (b) and (c) are identical representations. Images adapted from C. Colah (Colah, 2014). d) Example of a train-test split to solve for model parameters using a subset of the data and to test the accuracy of the fitting using the remainder of the dataset.

To better illustrate how the input vector and the labels are related, a toy example is shown in Figure 3.2 for a two class classification problem. In Figure 3.2 we seek to build a set of rules so that when given a set of $\boldsymbol{x}$ and $\boldsymbol{y}$ coordinates we can determine which label (either blue or red) the point should belong to. We do this by drawing a *decision boundary* between the blue and red scatter plots that will serve as a rule for classification. For this particular problem it is clear no linear decision rule will result in perfect classification accuracy. The decision boundary can either be non-linear (*tanh*) drawn on the input domain in Figure 3.2b, or it can be a linear decision boundary drawn on a non-linearly transformed input domain, Figure 3.2c. Both representations in Figure 3.2b and Figure 3.2c are identical. This is directly analogous to what is done in neural networks. By transforming the input space using linear operations usually in the form of an affine transformation followed by non-linear transfer functions such as *tanh*, a representation of the original data is created. This representation (Figure 3.2c) allows for much cleaner discrimination and easy decision rules can be made using this output space. Nesting such equations one within the other allows very complicated phenomena to be modelled. Such nested equations are in essence, a multi-layer neural network.

Many nested equations may be required to model the relationship between, for example, a 300 dimensional EELS spectra (input vector space) and a 3 dimensional oxidation state vector (output vector space). Finding the set of parameters that actually is able to map the input space to the output space (thus modelling quite complex phenomena) is difficult.

In a simpler case, a set of parameters (m and b) can be found to fit a line in the case of linear regression (Figure 3.2d). First a dataset is partitioned into two subsets, where one subset is used for *model training* and the other is used for *model testing*. During model training, a function is fitted to the data points and the degrees of freedom (m and b) are adjusted to fit the line. The remainder of the data in the test-set can be used to gauge how well the fitted line generalizes to data outside of its training set. This exact same approach is performed in neural networks training and testing and will be explained in detail in the following sections.

Recently, multilayer neural networks have become a popular universal function approximator for either regression or classification problems (Hornik, 1991). The most commonly used multilayer neural network architectures used today were developed by Hinton, Bengio and LeCun to solve a wide variety of problems ranging from computer vision to natural language processing to time-series prediction (Lecun, Bengio, & Hinton, 2015). In this study we exclusively use them as tools to perform spectroscopy analysis.

3.2 Operations in graph networks

During model training, the input data is propagated through the graph and the output neurons are compared with the labels to evaluate how well the current degrees of freedom are tuned to the classification or regression task at hand. This will be discussed in depth later. After model training when the degrees of freedom are fine-tuned, the model is ready for testing and is able to infer what a label is expected to be given unlabeled input data.

In a traditional multi-layer neural network, each dimension in the input vector is considered to be an input neuron. As the value of the input neuron propagates through the network from left to right, it is transformed through every hidden layer. The index notation seen in Figure 3.1 describes which neurons the weights connect. The jth neuron can be found in the current layer of neurons, i.e. layer (L), whereas k is used for indexing at layer $(L+1)$. Weight $w_{jk}^{(L)}$ is a scalar and when multiplied by the input into the neuron $a_k^{(L)}$ and summed with a constant $b_k^{(L)}$ it produces the output $z_k^{(L+1)}$ at layer $(L+1)$.

The total input into a neuron at index k of layer $(L+1)$ can then be represented by $z_k^{(L+1)}$ and it is summed over all neurons (J neurons total) from the previous layer indexed by j. This sum is shown in the arrows present between neurons in Figure 3.1.

$$z_k^{(L+1)} = \sum_{j=0}^{J} (w_{jk}^{(L)} a_k^{(L)} + b_k^{(L)})$$

Equation 2

To introduce non-linearities similar to what is shown in Figure 3.2, $z_k^{(L+1)}$is passed through a non-linear transfer function $\sigma(z_k^{(L+1)})$ which is also called an *activation function*. Non-linearities are a necessity since most classification problems are complex enough that a linear decision rule will not suffice (such as the toy example in Figure 3.2). Commonly used transfer functions include: tanh, sigmoids, softmax, and rectified linear units (ReLU). In the context of what is used in the present study only the ReLU is used as a hidden layer transfer functions and it can be formulated as the expression below. The choice in using ReLU as a transfer function, as opposed to other transfer functions, will be made apparent in the following section on *Back-propagation* when discussing automatic differentiation.

$$\sigma\left(z_k^{(L)}\right) = max(0, z_k^{(L)})$$

Equation 3

It is helpful to term this transfer function evaluated at some $z_k^{(L)}$ the *activation* or $a_k^{(L)}$ at layer L and neuron index k. The magnitude of this value determines the importance of this neuron. Combining the above equations together allows the activation of a neuron at layer $(L+1)$ to be evaluated based off of the previous input values at layer L across all neurons in the connecting layer.

$$a_k^{(L+1)} = \sigma\left(z_k^{(L+1)}\right) = max(0, \sum_{j=0}^{J} (w_{jk}^{(L)} a_k^{(L)} + b_k^{(L)}))$$

Equation 4

To better illustrate these operations graphically, refer to Figure 3.3 below.

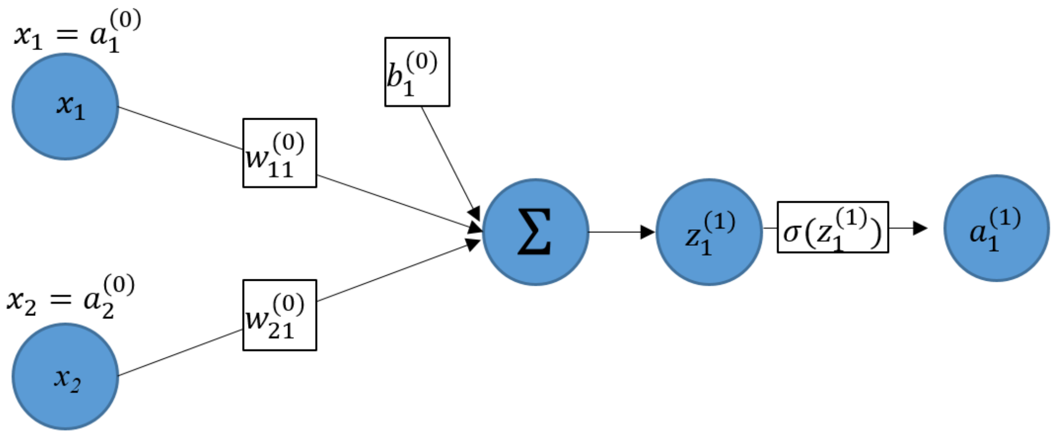


Figure 3.3. A step-by-step order of operations. The order of operations in **Figure 3.1** for one single hidden neuron in the first hidden layer. A dot product between the input neurons and the weight matrix is first summed with the bias to produce $\boldsymbol{z_1^{(1)}}$. Next, $\boldsymbol{z_1^{(1)}}$ is passed through the activation function $\boldsymbol{\sigma\left(z_1^{(1)}\right)}$ to produce the activation $\boldsymbol{a_1^{(1)}}$.

Combining the above equation with many hidden layers similar to the toy network shown in Figure 3.3 allows many nested equations with many free parameters to successfully map the input domain to the output domain. The success of multi-layer neural networks as function approximator is built upon this series of nested equations whereby the neural network, given enough hidden units, can approximate incredibly complex phenomena.

3.3 Using gradient descent to tune models (model training)

The training or "fine-tuning" of the network consists of finding the optimum values of all weights and biases in the network such that the network learns to correctly identify the label (e.g., correct valence) given an unknown input vector (e.g., spectrum). Initially, all weights are initialized by sampling from a distribution such as a Gaussian distribution. Biases are typically initialized at 1.00. The process of model training is two-fold. First, a forward pass is performed using the initialized parameters and evaluated to produce a set of predicted labels for each given input vector. Second, the network compares what the real label of the data was with what the predicted label was. At the start of the model fitting, the predicted label (with randomly initialized degrees of freedom) will output a trivial random guess. We can formulate the total error of the parameter configuration θ using a distance (loss) function between the real label and the predicted label.

$$\theta = [w_1, b_1, w_2, b_2, \ldots]$$

Equation 5

The set of parameters θ contains all degrees of freedom (weights and biases) in the network.

The distance between the predicted label and the real label can be computed in several ways. Both a mean-squared error and a log-loss error (also known as cross-entropy loss) are commonly used cost (or loss) functions.

$$C(\theta) = \sum_n \sum_k y_k \log p(y_k | x, \theta) \qquad C(\theta) = \sum_n \sum_k \left(y_k - a_k^{(L)}\right)^2$$

Equation 6

The equations which describes the *loss* (error) is shown above (Equation 6) for k number of classes. The term class is defined here as some categorical target variable such as a discrete oxidation state of a transition metal. Where $C(\theta)$ is the cost, $y_k$is the true one-hot encoded vector label of each input vector n samples, and $p(y_k|x, \theta)$ is the probability distribution of the predicted label given the input vector and the parameter set containing all degrees of freedom.

$p(y_k|x,\theta)$ and $a_k^{(L)}$ can be describing the same thing depending on which transfer function is used in the final layer of the network. This will be discussed in Chapter 4. The total loss is then evaluated over the full training set of data, N.

The loss function $C(\theta)$ which is a function of the current parameter configuration θ is directly analogous to the energy of the system in more commonly understood energy-based models (driving forces, potential energy). A large loss (i.e. high energy) indicates that the prediction was not a close estimate of the real label. A small loss (i.e. low energy) indicates that the prediction was a very close estimate to the real label. Extending the analogy it is clear that when the predictions are equal to the labels near the end of training, that the system is at a state of equilibrium i.e. there is no longer a driving force and the cost (loss, energy) is at a minimum. Another way to view $C(\theta)$ is to consider it a *loss surface* with spatial dimensions for every parameter in θ. The gradient of this loss surface, with respect to the tunable parameter configuration, can be considered to be the driving force and will hint at what steps need to be taken to find an equilibrium where loss is minimized. The relationship between loss and energy will be better understood later when discussing how the gradient of this energy term (loss) is used to go from the initial high energy configuration into a minimum of the loss surface.

After performing a forward pass of the network and evaluating how well the current parameters are tuned, the process of assigning "blame" to which parameters were not helpful in the inference can begin. These unhelpful parameters will be iteratively adjusted by changing their magnitude in a direction that reduces the distance between the predicted and real label. This is formulated by finding the sensitivity of the cost function to a small adjustment of a single parameter. Formally this is indeed taking the gradient of $C(\theta)$ and, using the partial derivatives of the cost function with respect to every parameter, to determine what the next estimate of the parameter should be. The magnitude of the adjustment is typically a scalar η also known as the *learning rate* multiplied by the gradient. The learning rate represents how large of an adjustment to make or alternatively, how large of a jump is made on the loss surface between parameter configurations.

$$w_1 = w_0 - \eta \frac{\partial}{\partial w_0} C(\theta)$$

Equation 7

The *learning* in a network can be understood by the simple equation above (Equation 7). The original parameter w_0 will be adjusted by some factor $\eta \frac{\partial}{\partial w_0} C(\theta)$ to create the new weight w_1. This is the heart of gradient descent. By following the gradient of the error with respect to each parameter, an energy minimized parameter configuration can be found by iteratively stepping towards it using a learning rate.

In practice, there are many problems with gradient descent optimization. The driving force for optimization, after all, is dependent on the sensitivity of the loss function with respect to a change in parameter configuration. Local minima on the loss surface are inevitable and there are many strategies to avoid local minima and to find the global minimum. One such strategy is to use past gradients to boost the jump over small energy barriers in the form of a momentum term. Another such strategy is to use higher order derivatives. Other strategies vary the learning rate over-time, a so-called adaptive learning strategy. These will not be discussed or explored in detail in this study. Off-the-shelf optimizers were used in the study for finding parameters considering the dimensionality of EELS spectra are still orders of magnitude smaller than data such as images which require very complicated optimizers. For further reading about different gradient descent algorithms in the context of deep learning please refer to Bengio (Bengio, 2012).

3.4 Back-propagation and the chain-rule

To compute the gradient of the cost function with respect to the weights, Rumelhart, Hinton, & Williams developed the back-propagation algorithm also known as automatic differentiation (Rumelhart, Hinton, & Williams, 1986) . The goal of back-propagation is to use the error found at the final layer of the network and propagate backwards through each layer of the network to compute the sensitivity of the global loss function to each possible degree of freedom. The sensitivity of the global loss function $C(\theta)$ with respect to a single weight can be represented below,

$$\frac{\partial}{\partial w_{jk}^{(L)}} C(\theta)$$

Equation 8

Calculating this derivative can be made possible using the chain-rule,

$$\frac{\partial}{\partial w_{jk}^{(L)}} C(\theta) = \frac{\partial z_k^{(L)}}{\partial w_{jk}^{(L)}} \frac{\partial a_k^{(L)}}{\partial z_k^{(L)}} \frac{\partial C(\theta)}{\partial a_k^{(L)}}$$

Equation 9

The above expansion describes the chain-rule in the context of a weight found attached to the output neurons in a neural network with only one hidden neuron per hidden layer. More generally, a summed version of this chain-rule expansion can be described for multiple hidden neurons per layer.

The first derivative shown below in Equation 10 and is the derivative of the affine transformation with respect to the weight in question. This is simply the activation of that neuron which is the values of the inputs into that neuron.

$$\frac{\partial z_k^{(L)}}{\partial w_{jk}^{(L)}} = a_k^{(L)}$$

Equation 10

The second derivative shown below in Equation 11 is the derivative of the activation function with respect to the affine transformation. It is now clear why the ReLU is a popular choice in activation function. The derivative of the ReLU function is indeed the unit step-function shown below. This is a very nice property because it improves the speed of model training by reducing computational complexity. Quite simply having a zero as this term when the activation is less than zero will stop this gradient since all terms are multiplied together via the chain rule.

$$\frac{\partial a_k^{(L)}}{\partial z_k^{(L)}} = \begin{cases} 1 & , \; z_k^{(L)} > 0 \\ 0 & , \; z_k^{(L)} < 0 \end{cases}$$

Equation 11

The final derivative is the derivative of the cost function with respect to the activation at the output neuron $a_k^{(L)}$. In the case of a mean-squared error cost function the derivative will be,

$$\frac{\partial C(\theta)}{\partial a_k^{(L)}} = 2(y_k - a_k^{(L)})$$

Equation 12

The above formulation and derivatives are assuming that the weight in question is immediately attached to the output neurons. To compute gradients for neurons in other hidden layers, the formulation of the derivative will be a much longer chain of derivatives but the same chain-rule approach applies. If any neuron between the output and the weight being calculated contains a value of $z_k^{(L)}$ which is less than zero, the entire gradient will go to zero due to the properties of the ReLU. Gradients are calculated layer by layer starting from the output neurons and backwardly propagating until the final gradients are calculated of the weights connecting the input neurons, hence the naming *back-propagation*.

3.5 Convolutional neural networks

A special and more constrained architecture of a multilayer neural network is a *convolutional neural network*. Like the name suggests, instead of using an affine transformation as a parameterized linear transformation, these are replaced by *convolutions*. A convolution contains a *filter* (also called a kernel) which slides across the data and performs a dot-product between the data and the weights in the filter.

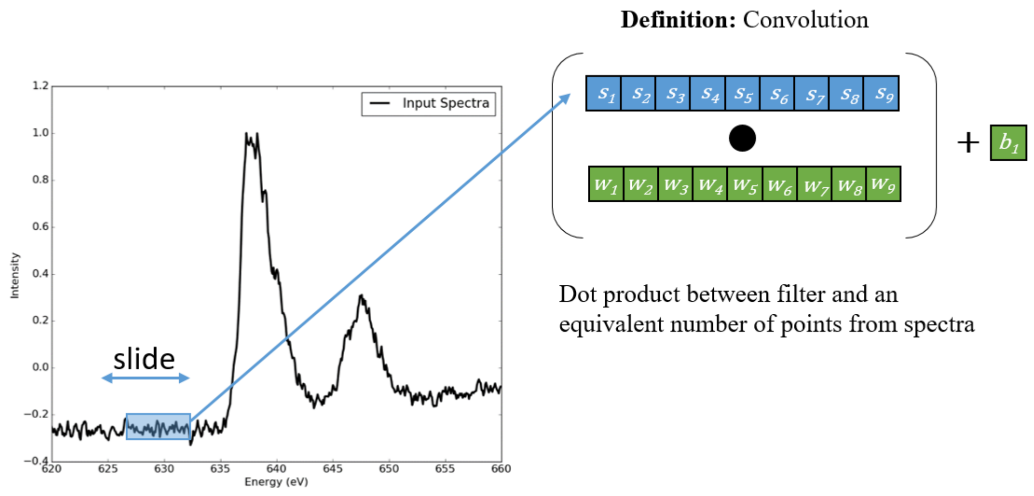

Definition: Convolution
Input Spectra
slide
Intensity
Energy (eV)
s_1 s_2 s_3 s_4 s_5 s_6 s_7 s_8 s_9
w_1 w_2 w_3 w_4 w_5 w_6 w_7 w_8 w_9
$+$ b_1
Dot product between filter and an equivalent number of points from spectra

Figure 3.4. An example of a convolution kernel sliding over an input spectrum.

Before we can begin to understand what advantages a convolutional neural network may have when compared to a multilayer neural network, let us consider the type of data that we frequently encounter in spectroscopy. Spectroscopy data is ordered time-series data. Some parts of the spectrum are correlated with each other. Put simply, when some bins of a spectrum have a large number, some other bins of the spectra are more likely to have a large number. It would be advantageous for a statistical model to take into account local correlations between adjacent dimensions of the input vector. If a proposed statistical model could take into account the *ordering* of the bins of the spectrum, then clearly it would need less parameters to create a successful classification model.

A multilayer neural network, which we will hereafter refer to as a *dense* neural network, performs a point-wise affine transformation for every single bin of the spectra because it assumes every bin is totally uncorrelated with no ordering. The choice of naming a multilayer neural network a dense neural network is now clear – the weights connecting neurons are densely connected. That is to say that every combination of hidden layer neurons and input neurons are explored. This is not the case in a convolutional neural network. Instead, a hidden neuron uses a shared weighting scheme where the same weights slide over the spectra in the form of a convolution kernel to take into account local ordering. An illustration of this shared weighting scheme for one hidden neuron is shown in Figure 3.4. In a dense neural network, there is one weight per hidden neuron. When considering a convolutional neural network, we have a vector (kernel) of weights per hidden neuron.

In Figure 3.4 a kernel of length 9 is being convolved with 9 indices of the input data at a time before being summed with a bias term. This linear transformation replaces the affine transformation used in dense neural networks. The output of this neuron will be physical and representable as a convolved version of the original data.

Each dimension in the convolution kernel will contain a trainable weight. Formally the convolution can be written in a similar manner to the affine transformation which we were introduced to in Chapter 2.2.

$$z_j^{(L+1)} = \left(\sum_k w_k * a_k^{(L)}\right) + b_j$$

Equation 13

A convolution (i.e. a sliding dot product) denoted by the $*$ operator between the input neurons and a trainable set of weights w_k for a given hidden neuron j in layer $(L+1)$ is shown above in Equation 13. This contrasts the affine transformation where the weight matrix $w_{jk}$is unique between every input neuron and hidden neuron j. It is clear that a convolutional neural network will have significantly less free parameters than a dense neural network since the length of the convolution kernel w_k is fixed and not dependent on the input size. When considering back-propagation and model training, everything still applies from the previous sections on dense neural networks.

4 Towards calibration-invariant spectroscopy using deep learning

4.1 Overview

Statistical learning methods in spectroscopy have been slowly percolating the literature for the last couple decades. Gallagher and Deacon, in a pioneering study in 2002, used single layer dense neural networks to predict experimental X-ray spectra for the automated classification of minerals (M. & P., 2002). Timoshenko *et al.* used multi-layer dense neural networks to predict the coordination numbers in metallic nanoparticles given a full simulated x-ray absorption spectroscopy spectrum (Timoshenko, Lu, Lin, & Frenkel, 2017) and Zheng *et al*. used an ensemble-learned matching scheme to characterize simulated x-ray absorption spectroscopy generated from the Materials Project where one of the steps explicitly involves peak shifting (Zheng *et al.*, 2017). Carey *et al*. used unsupervised methods such as nearest neighbor clustering approaches and studied the effect of preprocessing on the classification of Raman spectra (Carey, Boucher, Mahadevan, Bartholomew, & Dyar, 2015). Lopez-Reyes *et al*. developed unsupervised (PCA) and supervised methods (dense neural networks) to classify minerals on the ExoMars rover with Raman spectroscopy (Lopez-Reyes, Sobron, Lefebvre, & Rull, 2014).

For the classification of the entire Raman spectroscopy database RRUFF, Liu *et al*. used a convolutional neural network feature extractor (Liu *et al*., 2017). Their methodology was two-fold. First a convolutional neural network feature extractor, then for classification, a dense neural network was used. They compared these results to other common machine learning classifiers such as boosting, random forest, and support vector machines. To teach the network to understand small translational (chemical) shifts, they used data augmentation with small random crops/shifts of the spectra.

Convolutional neural networks have not been used presently in electron spectroscopy. Yedra *et al*. created a script in Digital Micrograph, named *Oxide Wizard*, to differentiate the oxidation states for various metal oxides in EELS (Yedra *et al*., 2014). Using domain knowledge they engineer features such as the ratios of peaks, the FWHM of peaks and the distances between oxygen and the metal. Zhang *et al*. used a multiple linear least-squares technique to determine the valence of Mn in EELS (Zhang *et al*., 2010). With reference spectra, they fit a generalized linear model using known reference spectra. Tan *et al.* also showed a comparison between common methods in EELS to determine the oxidation state, such as least-squares fitting and feature engineering using peak-ratios. They determined that different methods are ideal for different transition metals (Tan *et al*., 2012).

The outlined methods used in the past for oxidation state classification in EELS are based on either manual feature extraction (i.e. ratio of peaks) or rely on the pattern matching ability of human spectroscopists to observe similarity between references and unknown spectra. The pattern matching ability of the human brain has been replicated here using recent advances in deep learning with convolutional neural networks originally designed for self-driving cars and many object recognition tasks. We develop here a challenging dataset to be used as a test-case to probe the effectiveness and generalizability of convolutional neural networks in spectroscopy classification. As a model system, we investigated the valence identification of Mn, this being relevant in many fields of materials research, particularly interesting in the study of valence in battery materials. We acquired 2001 Mn^{2+}, Mn^{3+}, and Mn^{4+} EELS spectra to be used as a model training and model validation dataset and we also digitize 31 Mn^{2+}, Mn^{3+}, and Mn^{4+} reference spectra from published articles to be used as a test-set. Our spectroscopy optimized convolutional neural network architecture, named SpectralNet, was trained and tested on the data and is able to discriminate between the very qualitatively similar spectra of three different valences of Mn with high accuracy. When considering the differences in calibration between different instruments, the peak onset is even more challenging to compare in electron energy loss spectroscopy. SpectralNet, however, is capable of these types of classification tasks and is proven to be one optimal convolutional neural network architecture for electron spectroscopy with generalizable applications to any field of spectroscopy.

The ideal spectroscopy-optimized statistical classifier is one that is able to infer what the correct valence is from an unknown signal. It can achieve this by remembering what previous data it had looked at. This is called model *training*, and for the classifier to accomplish such a goal requires four steps (Figure 4.1 contains a flow chart describing the steps).

1. Data Collection: A large set of data needs to be collected that adequately encompasses the variance in the type of unknown spectra that may be presented to the classifier after model training.
2. Pre-Processing: The data in EELS is in the form of a hyperspectral spectrum image. Not all pixels of the image will actually be the Mn compound we are interested in and the carbon TEM grid substrate measurements are removed. In addition further steps like normalization are required.
3. Model Training: Tweaking the free parameters of the model such that they are able to predict the outcome in the training set (sub-set) of the data.
4. Model Testing: The model will also need to be tested rigorously using a variety of metrics to determine how well the classifier will perform in a real-world setting.

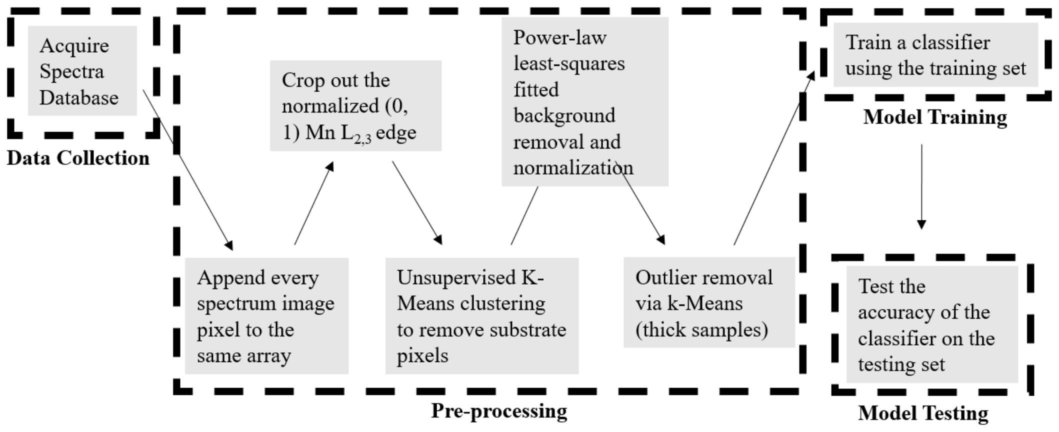

Acquire Spectra Database
Data Collection
Append every spectrum image pixel to the same array
Crop out the normalized (0, 1) Mn $L_{2,3}$ edge
Unsupervised K-Means clustering to remove substrate pixels
Power-law least-squares fitted background removal and normalization
Outlier removal via k-Means (thick samples)
Pre-processing
Train a classifier using the training set
Model Training
Test the accuracy of the classifier on the testing set
Model Testing

Figure 4.1. Work-flow of going from data collection to model testing and all of the steps in between. Each of these steps will be discussed in depth in the preceding sections of Chapter 4.

4.2 Data Collection

A total of 4979 electron energy-loss spectra of Mn^{2+}, Mn^{3+}, and Mn^{4+} were acquired using an FEI Titan transmission electron microscope in a variety of conditions. After preprocessing and identifying clean data, the final dataset is comprised of 2001 spectra in total. This includes

448 Mn^{2+} spectra, 765 Mn^{3+} spectra, and 788 Mn^{4+} spectra cropped between 635 and 665 eV (300 bins at a dispersion of 0.1 eV/bin).

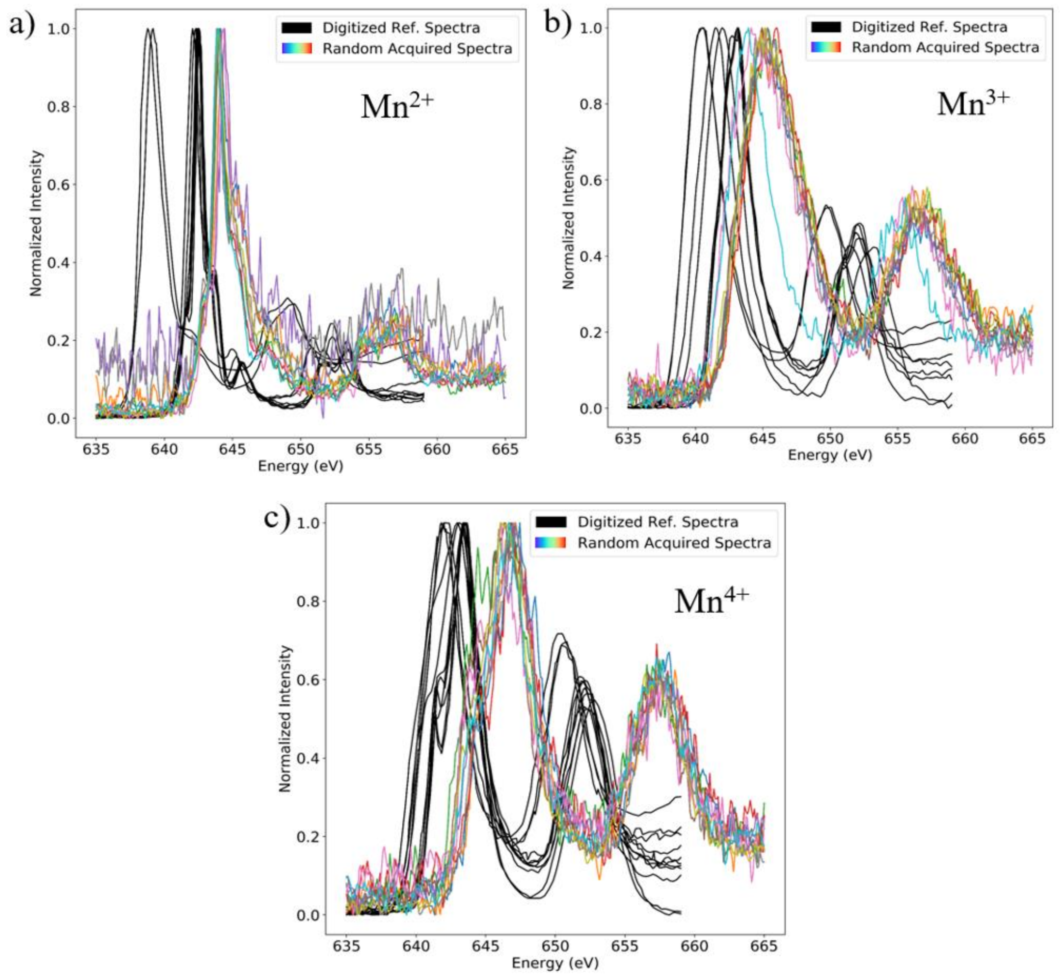


Figure 4.2. a) 12 digitized reference spectra of various Mn^{2+} compounds taken from published articles (black). 12 randomly selected MnO (Mn^{2+}) spectra acquired for this study. b) 10 digitized reference spectra of various Mn^{3+} compounds taken from published articles (black). 10 randomly selected Mn_2O_3 (Mn^{3+}) spectra acquired for this study. c) 9 digitized reference spectra of various Mn^{4+} compounds taken from published articles (black). 10 randomly selected MnO_2 (Mn^{4+}) spectra acquired for this study. All digitized spectra shown are taken from digitizing the figures in Garvie *et al.*, Zhang *et al.* and Tan *et al.*

To obtain spectra from a wide variety of instruments and resolutions to prove generalizability, reference spectra were digitized from three studies by Garvie *et al.*, Zhang *et al.* and Tan *et al.* (Garvie & Craven, 1994; Tan *et al.*, 2012; Zhang *et al.*, 2010). These spectra were not used for training SpectralNet and are instead used as a test-case representing signals that are well outside of the distribution of the acquired data. They are of significantly higher resolution and the differences in instrument calibration is clear, with the onsets of the peaks being different as shown in Figure 4.2. The energy range of the digitized spectra is between 635 and 658 eV due to many of them being cropped for their respective publications.

A qualitative inspection of Figure 4.2 highlights that Mn^{2+} is narrower than both Mn^{3+} and Mn^{4+}. In addition, Mn^{4+} can be differentiated from Mn^{3+} by a small shoulder located on the low energy side of the larger peak.

All samples were acquired using a monochromated FEI Titan 80-300 transmission electron microscope (TEM) operating at an accelerating voltage of 80 keV. The FWHM of the zero-loss peak was ~0.2 eV. Reference Mn oxide samples above 99% purity (obtained from Sigma-Aldrich) were crushed between SiO_2 glass slides to produce a fine nanoparticulate powder. A holey carbon TEM grid was tapped onto the glass slide to adsorb fine particulates. In STEM mode, spectrum images were collected over a 2D area on the edges of the surfaces of the nanoparticulate Mn oxides to capture the thinnest areas. A spectrum image is a collection of spectra acquired at each pixel of a two-dimensional area containing the particulates. 10, 10 and 15 spectrum images of MnO, Mn_2O_3 and MnO_2 were acquired for a total of 1604, 1512 and 1863 individual spectra respectively. Spectrum images were acquired on a variety of thicknesses and crystallographic orientations to reduce anisotropic effects by introducing a wide variance of samples. Refer to the figure below to see the ADF images of what the samples looked like on the TEM grid.

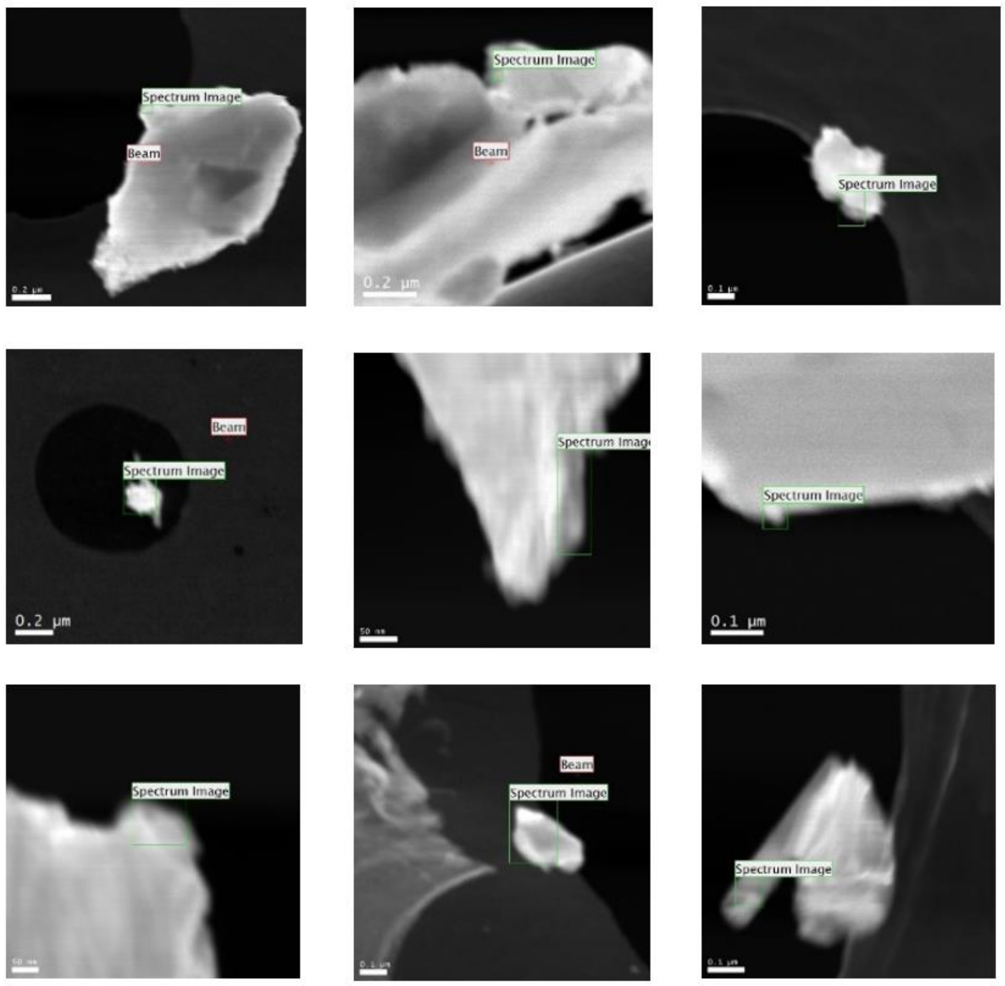


Figure 4.3. Angular Dark Field (ADF) images of some of the samples showing varying sizes of Mn oxide nanocrystals. Acquisitions were carried out on the edges of most samples to get thin samples.

4.3 Pre-processing

Every spectrum image was unzipped and appended to the same array per valence state using the Python library *HyperSpy* to open the *Digital Micrograph* .dm3 files. The EELS spectra were then spectrally cropped to a length of 300 histogram bins (dispersion of 0.1 eV/channel) near the Mn $L_{2,3}$ core-loss edges so that only Mn $L_{2,3}$ edge data was included.

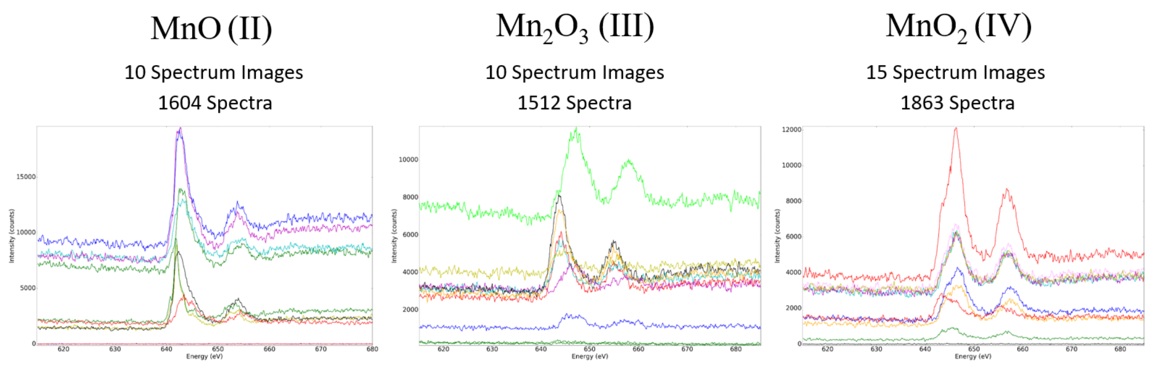


Figure 4.4. 10 randomly sampled raw spectra per valence of each of the three valences of Mn. Very thick samples (those with low counts) are shown to be very featureless and low quality. Thick samples are considered to be out of the specifications of the microscope and need to be removed.

To remove pixels containing only substrate signals from the coalesced dataset, k-Means clustering was performed on each subset of the data (Mn^{2+}, Mn^{3+} and Mn^{4+}). By qualitatively inspecting the cluster center, it is apparent which cluster center belongs to the Mn $L_{2,3}$ core-loss edges and which cluster center belongs to the substrate (where no $MnL_{2,3}$ edge is present) . The cluster centers are shown in Figure 4.5. After removing substrate pixels, there were a total of 1273, 1342 and 1687 Mn oxide spectra for Mn^{2+}, Mn^{3+} and Mn^{4+} respectively.

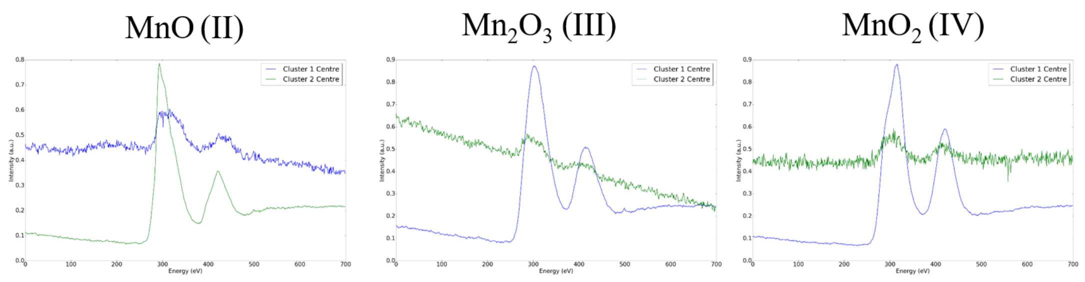


Figure 4.5. Example of the unsupervised learning technique K-means clustering being used to separate the Mn oxide samples from the carbon substrate spectra assuming two cluster centers. Spectra belonging to the cluster center that looks like substrate were removed.

To remove the background prior to the onset of the core-loss edge, a generalized power-law was least-squares fitted to the indices prior to the edge onset and subtracted from each spectra. Next, each spectrum was individually normalized between 0.0 and 1.0 and mean-subtracted. The choice of these preprocessing steps was empirical and determined via cross-validation (discussed later).

There is a sizeable portion of low quality data present in the dataset as a result of trying to get a wide range of varying thickness and crystallographic orientations in the samples. If samples are too thick then they are not electron transparent and no ionization edge is visible. Typically only samples under 100 nm are thin enough for the beam to transmit through. To remove these low quality spectra, an iterative k-Means clustering approach was used. Each class of Mn was clustered with two or three cluster centers and through qualitative inspection of the cluster center mean, thick and thin samples could be discriminated.

4.4 Data augmentation

The robustness of any statistical classifier is entirely derived from the input data being able to successfully explain the variance found in real world samples. This can be artificially simulated by using data augmentation schemes which apply transformations on the input data to increase variance.

To artificially inflate the number of training data, a data augmentation scheme was used to increase the model's robustness to the noise frequently found in EELS. To model the noise found in EELS signals, scalar multiples of low variance principal components are linearly combined with each training example. Principal components analysis has grown increasingly popular at denoising EELS spectra recently (Bonnet & Nuzillard, 2005; M. Bosman *et al.*, 2007; M. Bosman, Watanabe, Alexander, & Keast, 2006; Michel Bosman, Keast, Watanabe, Maaroof, & Cortie, 2007; Cueva, Hovden, Mundy, Xin, & Muller, 2012; Roth, Kaeberle, & Hsu, 1982). Removing low variance principal components is effective at eliminating some types of noise frequently detected in EELS with minimal loss of signal (typically $< 10^{-2}$ % of the signal is removed). Using these low variance principal components as a noise distribution, they were added to each input spectrum in scalar multiples ranging between zero and five times the baseline noise level. In this context, a scalar multiple of zero would be PCA cleaned data (removing low variance components), and a scalar multiple of 1 is the original signal.

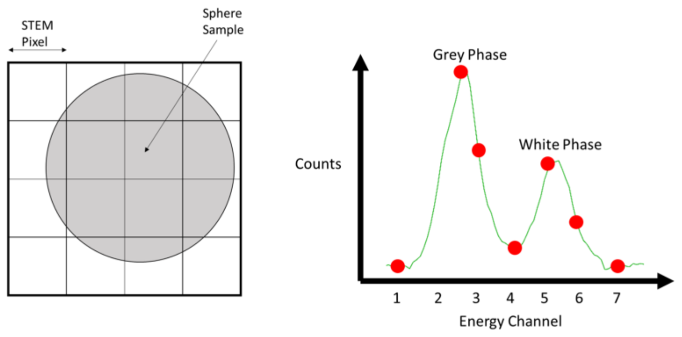
4.4.1 Principal Component Analysis

Principal component analysis (PCA) is one commonly used dimensionality reduction technique when dealing with higher dimensional data. PCA is growing in popularity in the electron microscopy community due to its effectiveness in denoising images and spectra with minimal loss of signal. Dimensionality reduction is performed by re-projecting the original data along the direction of largest spread found in the set of data. Subsequent directions are found that also explain the variance on the condition that they are also orthogonal to the original direction of largest spread. In determining these orthogonal directions of spread, which are also known as principal components, it is possible to then determine what is the minimum number of principal components that would be required to describe the majority of the signal.

Formally, PCA is performed on a dataset containing N observations of M features. This can be represented as an array X of size N by M. In the context of electron energy loss spectroscopy, an observation would be a single full spectrum or a pixel from a spectrum image. Each feature would be an energy histogram bin. The length of M in this context is the number of channels available on the spectrometer.

A qualitative example of how a traditional hyperspectral spectrum image is interpreted in statistical learning is shown in Figure 4.6. In Figure 4.6a, a toy example of a 7 channel 4x4 spectrum image is shown. The summed spectrum image (Figure 4.6a, right) shows the full signal summing across all pixels. When observing pixels one by one, we can split the spectrum image into 16 (4x4) separate signals (also known as an observation) of a length equivalent to the number of energy channels (i.e. features).

a)



b)

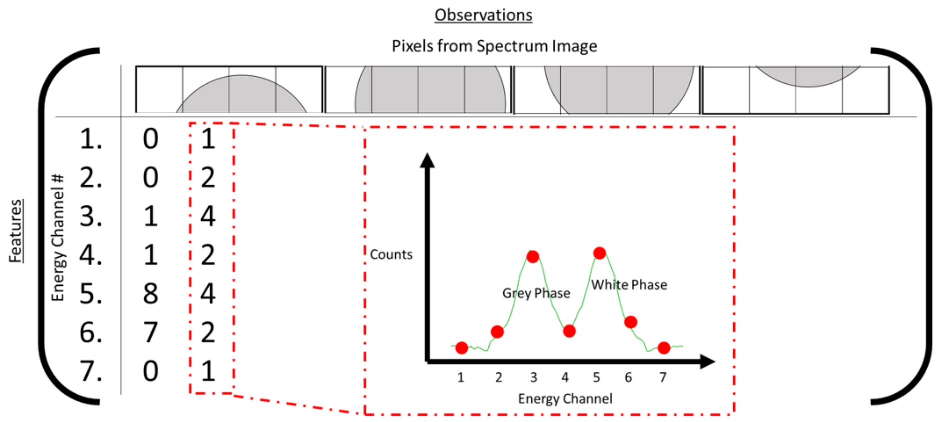


Figure 4.6. a) An example of a 4x4 spectrum image taken of a grey sphere on a white substrate. The summed spectrum image on the right explains how the grey phase is collected in the first few energy channels, while the white phase is collected in the last few energy channels. B) An example of forming a second rank matrix of observations and features using a spectrum image.

Formulating a second rank matrix X of the form N by M causes all of the familiar algorithms which can manipulate linear algebra apply to these types of data. The covariance of matrix X can be calculated which is a measure for how correlated any two given features are to each other. In other words, the covariance can tell us how an increase in the magnitude of one feature is coupled with an increase in magnitude in another feature. Exploiting the covariance matrix of X is the central idea behind PCA. The covariance matrix of X will be a second rank matrix of size M by M where the covariance is calculated between every combination of feature found in the dataset.

Formally, the covariance between a feature vector M_1 of observations and a feature vector M_2 of observations can be described as,

$$Cov(M_1, M_2) = E[\,(\,M_1 - E(M_1)\,)(\,M_2 - E(M_2)\,)\,]$$

Equation 14

Where $E(x)$ is the expectation (arithmetic mean) of feature vector x. The resulting covariance matrix then can be considered to be a representation describing the correlations of how the features interact with each other. Another way to think about the covariance matrix, is that it is a method to describe the variance in cases where you have more than two variables.

Using the covariance matrix, one can then find the eigenvectors and eigenvalues using decomposition algorithms. Performing a decomposition on a covariance matrix allows the principal components of a dataset to be found. Intuitively solving for these values allows the direction of largest spread to be found.

As an example, imagine a spectrum image, which is a collection of EELS spectra taken in a grid pattern, taken on a homogenous sample. In an ideal acquisition, all pixels of the spectrum image (i.e., all spectra) should be identical no matter which location on the sample the spectra was collected. In practice however, there will be noise and other differences between the spectra of the entire set (spectrum image) of spectra. PCA can be used to find what is common among the set of spectra. In this manner, provided there are enough examples, the signal can be separated from some types of noise using the principal components. An explained variance ratio can be used to describe the principal components and *how much of the signal* that a single principal component is capable of describing. This explained variance ratio is a ratio of the eigenvalues for each respective eigenvector and is typically displayed in a construction termed the *scree plot*. Refer to the figure below to understand how a single spectra can be denoised assuming it is part of a spectrum image.

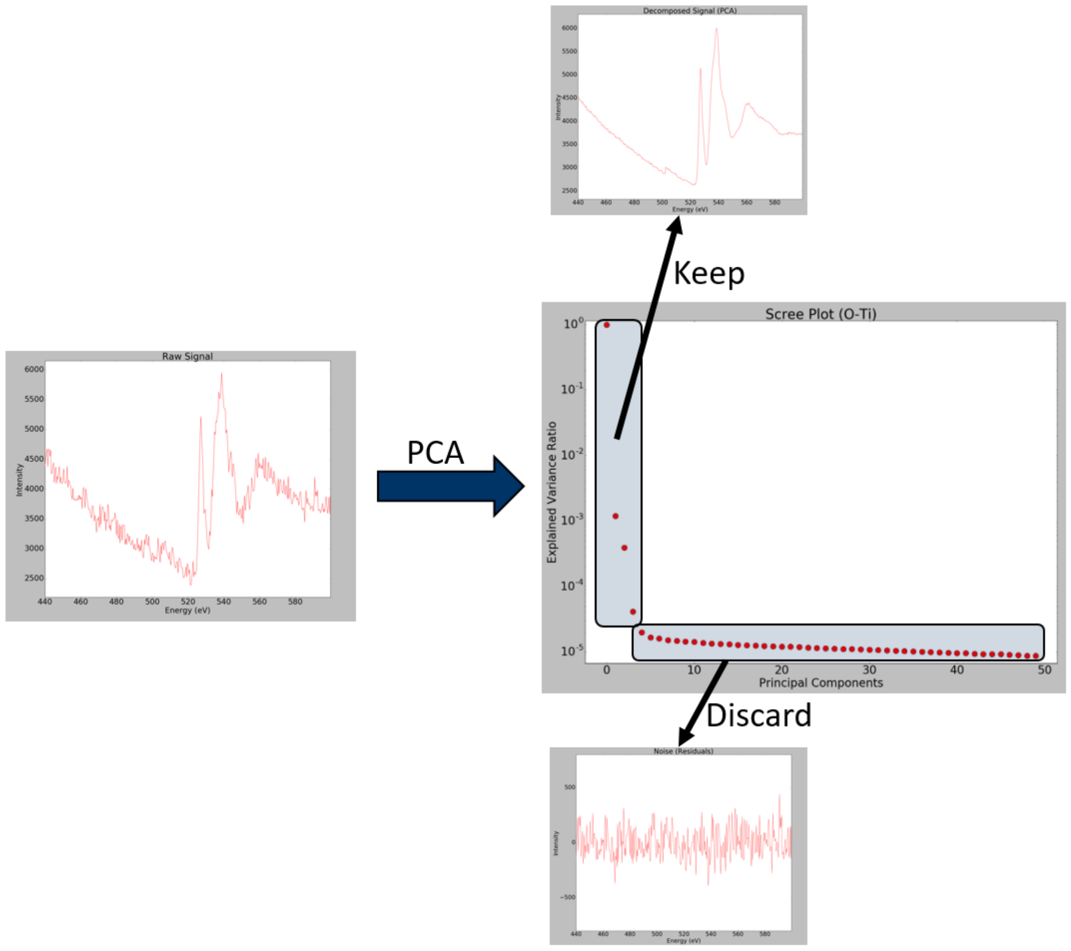


Figure 4.7. An example of PCA being used to denoise an oxygen K edge. High variance principal components contain parts of a spectra that can explain most of the signal and low variance principal components are typically noise. The discarded noise is in fact the signal from the 5th through 50th principal components.

4.5 The SpectralNet architecture

After data augmentation and pre-processing, a large dataset now is finalized to be used to train a neural network. These spectra in addition to their labels are used to find the free parameters in a specified neural network architecture.

Directed graph networks, such as a convolutional neural network, operate on the results of previous layer activations as an input is propagated through the network. In this way, it is only the very first hidden layer that interacts with the original input spectra and subsequent operations are performed on the convolutions and activations from the previous layers. SpectralNet is structured in a triangular and hierarchical manner where the first filters make a coarse decision boundary of where a large discriminating difference exists between the classes (i.e. oxidation states of Mn). By performing subsequent operations on the results of previous layers, more granular differences will be found until the spectra is abstracted into a feature vector embedding of the original input specifically optimized for discriminating between classes.

The architecture of SpectralNet consists of 5 feature-extractor blocks and 1 classification block. The 5 successive feature-extraction blocks project the 300-length input vector spectra onto a space that allows easy fingerprinting and discrimination between the different classes. Using these lower dimensional features, the classification task to determine each of the different classes can be performed using the classification block.

A schematic representation of the neural network graph of SpectralNet is shown in Figure 4.8. In this schematic diagram, the edges and neurons in each feature extraction block represent first a linear convolution, then activation via a rectified linear unit (ReLU) transfer function, a batch-normalization, and finally average pooling. Within the classification block, the 12 features of length 6 are convolved (with dropout regularization) and batch-normalized before being collapsed using global average pooling (GAP) and activated using multinomial logistic regression (the softmax function) to produce predicted class probabilities.

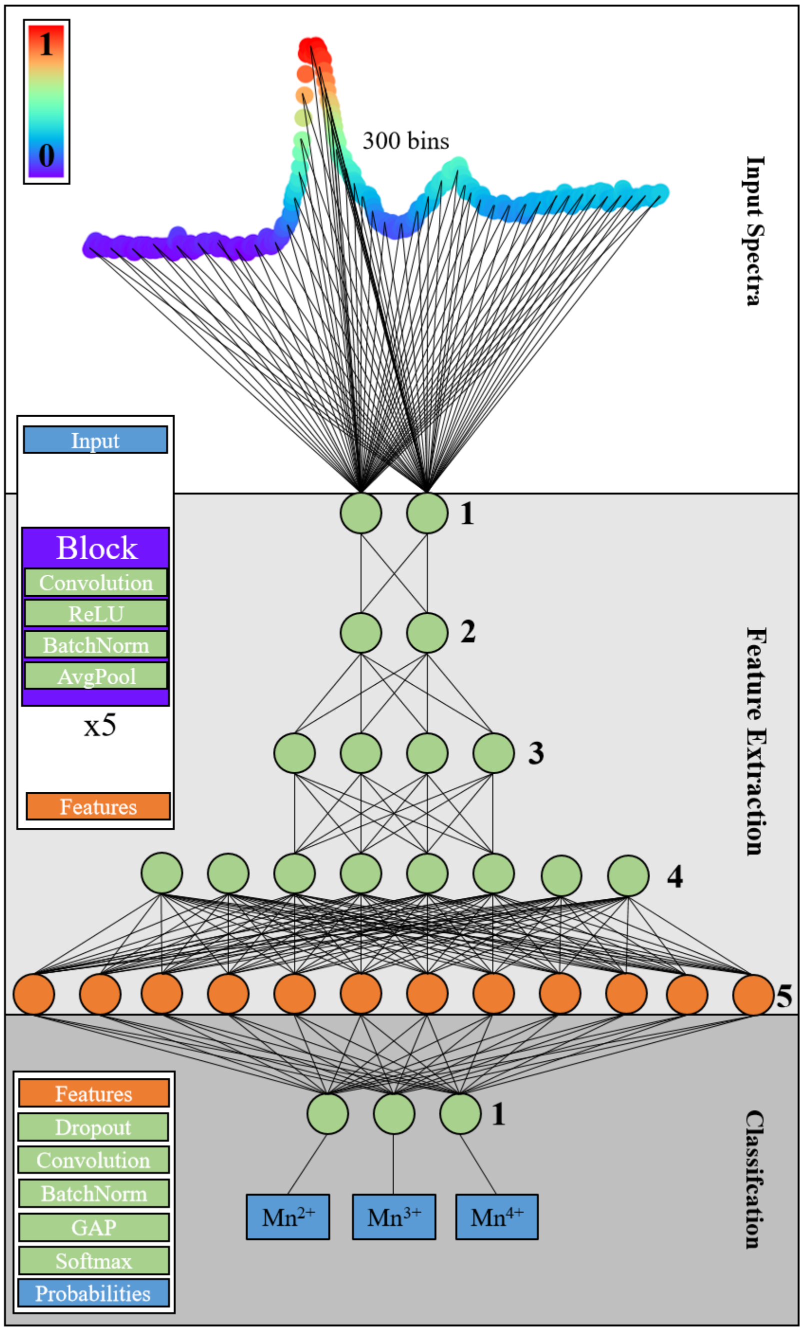


Figure 4.8. Schematic of the neural network graph of SpectralNet. The input spectrum is first passed through the 5 successive feature extraction blocks where each block contains the operations found in the purple rectangle. The output at the 5th block (orange neurons) is considered to be a representation of the original input data that is optimized for discrimination (i.e. discriminative features). These features are then used to classify the valence of the inputted spectrum using the single classification block.

After SpectralNet is fully trained on all of the training data, it is possible to observe which transformations were chosen that allow a clear separation between classes. The output at each neuron of the graph in Figure 4.8 is shown in Figure 4.9. The same set of operations with the same parameters are performed on each spectrum but it can be observed that the output at each block is indeed different between valence states. This is due to the network choosing convolutional filters during training that are highlighting the differences found in all three classes of input spectra. These differences become magnified as they are propagated through the network since gradient descent was used to tune and optimize these convolutional filters for class discrimination. The final feature extractor block creates a representation of the original input spectra that is exclusively optimized for differentiating the three valences of Mn. The classification block then mixes-and-matches, via a depth-wise convolution, which of the 12 activations are unique to each valence. One unique filter is shown to activate strongly for each unique valence.

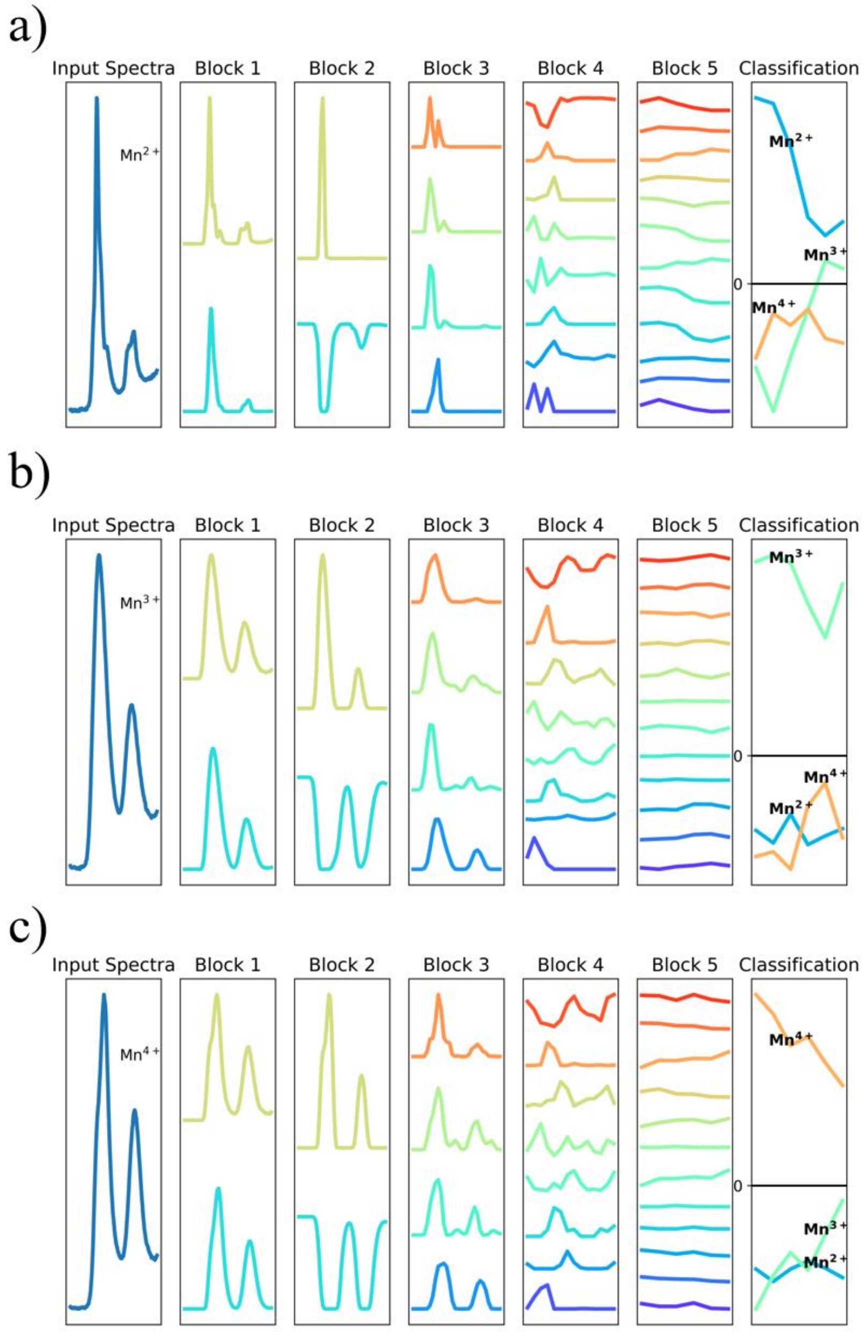


Figure 4.9. a-c) Activation outputs per feature extraction block and classification block averaged over all training examples of Mn^{2+}, Mn^{3+}, and Mn^{4+} respectively. The input spectra shown is the average of all spectra in each valence.

4.5.1 Feature extraction architecture

Each feature extraction block in the SpectralNet architecture has first a 1D convolution, activation using a rectified linear unit (ReLU), followed by normalization, then finally an average pooling layer (i.e. 2x averaged down-sampling). These operations are shown in Figure 4.10.

The input into each feature extraction block is convolved with a sliding (stride of 1) kernel of length 9, 7, 7, 5, or 3 for the 1st through 5th blocks respectively. The amount of filters per block gradually expands from 2, 2, 4, 8 and 12 filters for the 1st through 5th blocks respectively. The choice for the amount of filters per block was inspired by the feature extractors used in the ImageNet competition like the well-established AlexNet and VGG16 architectures (Krizhevsky, Sutskever, & Hinton, 2012; Simonyan & Zisserman, 2014) . The convolutions in a convolutional neural network can be considered to be *depth-wise*. For example, each convolution filter from the 2nd block will be convolving both activations in the 1st block at the same indices as the kernel slides across the spectra. The activation (output) maps for each filter is shown in Figure 4.9.

After convolution, activation occurs using a ReLU transfer function which is used to draw a decision boundary and to add non-linearities. ReLU transfer functions are one of the most popular transfer functions used in the literature due to the trivial derivative which aids in reducing computation time during gradient descent.

Refer back to Chapter 3 for a description of this style of convolution and the non-linear transfer function the ReLU in addition to gradient descent and back-propagation.

To re-normalize the data between feature extraction blocks, a per-batch normalization is performed to mean-center the data and to standardize the data to unit variance. The data is originally preprocessed to be normalized before model training. However after subsequent convolutions and activations, this will no longer be the case. This phenomenon is called the internal covariate shift and this batch normalization procedure developed by Ioffe and Szegedy is a commonly used method for normalization between operations in many popular neural network architectures (Ioffe & Szegedy, 2015).

In an effort to reduce the amount of noise in the data and also for dimensionality reduction, at the end of every feature extraction block a down-sampling (average pooling) was performed. Average pooling, or max-pooling, is a well-established technique used in many neural network architectures for dimensionality reduction and for creating a feature vector.

4.5.2 Classification architecture

The architecture shown in Figure 4.11 is shown to be in two parts. The feature-extraction half of the architecture creates a representation of the input spectrum which allows cleaner discrimination between classes. The second half of the architecture, the classification half, is responsible for drawing this decision boundary.

To keep the model lightweight and with minimal parameters, no dense layers (*i.e.* fully-connected artificial neural network hidden layers) were used. The concept of dense layers was introduced in Chapter 3. Instead, the classification block consists of a drop-out layer, three depth-wise convolutions with no activation, global average pooling, then finally activation with a softmax function. This dense-layer free architecture has been used in recent architectures such as MobileNet, and SqueezeNet to produce light-weight architectures (Howard *et al*., 2017; Iandola *et al*., 2016). This dense-free architecture also significantly increases generalizability in the case of temporal shifts in time-series data which we prove later in Figure 4.14.

Overfitting can occur when the network has learned to memorize the training distribution and no longer can generalize to the test distributions. This is analogous to using a high order spline fit to model a distribution where a linear line would suffice. Extrapolation outside of the distribution would lead to significant error. One commonly used technique to limit large amounts of overfitting is to use the stochastic method of neuron dropout developed by Srivastava *et al* (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014). Dropout is used to prevent overfitting by randomly selecting a percentage (we use 80%, found using cross-validation) of weights per batch between the weights connecting the feature extractor neurons and the classifier neurons and setting them equal to zero. Dropout is used so that during model training the optimum parameter configuration will be one that does not heavily rely on a small subset of parameters. This also greatly increases the difficulty of the classification task which also aids in preventing overfitting.

The output of the feature extraction blocks are 12 feature vectors which represent a transformation of the input spectra into a feature space that allows easier fingerprinting and discrimination. Three neurons, one for each of the classes, pick and choose which combination of the 12 feature vectors are unique to each class. These neurons are considered depth-wise convolutions and use a dot product (plus bias) with trainable weights to select the right combination of feature vectors. Global average pooling (GAP) will collapse this final convolution layer into one averaged activation per filter where the magnitude of this number describes how strongly the filter activated. In other words, if the depth-wise convolution filter found a spectrum with a feature fingerprint similar to the examples it saw during training, then the averaged value of the filter will be large.

To convert the final neuronal activations into a predicted class probability for classification, the activations are passed into a logistic regression function, also known as the softmax activation function. The magnitude of the GAP layer for each of the three classes is used as input. The softmax function converts the three magnitudes for each of the three filters into probabilities where all probabilities sum to 1.00. The choice of softmax is commonly used in many architectures because it is easily differentiable, a requirement for the back-propagation algorithm.

$$p(y_k|z,\theta) = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

Equation 15

The predicted class probability $p(y_k|z,\theta)$ of the correct label y_k, given a set of parameters θ with an input spectra feature vector z_j, is shown above in Equation 15. This softmax function (i.e. generalized multinomial logistic regression) is the final layer in the SpectralNet architecture and is a commonly used classification operation used in neural networks.

4.6 Model Training and Model Validation

4.6.1 Model Training

Stratified 10-fold cross-validation was used for model validation and to estimate the error of the withheld test set. The acquired Mn dataset was divided into 10 roughly equal *folds* (partition) where each class is *stratified*. This means there was an even proportion of each class in each fold. 9 folds (i.e. 90% of the data) was used for finding model parameters while the last fold was used for calculating validation-set accuracy. This was repeated for every fold in the model to produce 10 models trained on different partitions of the data. The training set was augmented by a factor of 60 by using principal components analysis (PCA) as a tool to add or remove noise from the data. A description of the data augmentation routine can be found in the *Data Augmentation* section.

In batches of 2048 randomly selected augmented training spectra at a time, the spectra are non-linearly transformed over all edges and nodes in the directed graph to produce predicted class probabilities $p(y_k|x,\theta)$ for all possible classes. The cost function $C(\theta)$ is minimized between the true one-hot encoded label y_k and the output at the output neurons (described by the predicted class probability $p(y_k|x,\theta)$). This is calculated over all *N* training examples in the batch and for all possible *K* classes. The appropriate cost function used with a softmax activation function that produces probabilities is the cross-entropy (Equation 6, left). The error from the cost function can be back-propagated backwards using the chain-rule through each weight and bias in the network to assign blame as to which parameters were unhelpful in the classification. With this scheme, the weights and biases can be nudged in a direction that minimizes error using stochastic gradient descent (SGD). The specific SGD algorithm used was the adaptive moment estimation optimizer, the Adam optimizer commonly used for the training of neural networks (default parameters) (Kingma & Ba, 2014).

In Figure 4.12 the training accuracy as a function of *epoch* is shown. An epoch refers to the gradient descent iteration after doing a full pass through the data augmented training data. The accuracy gradually improves over time and the loss smoothly approaches zero after 200 epochs.

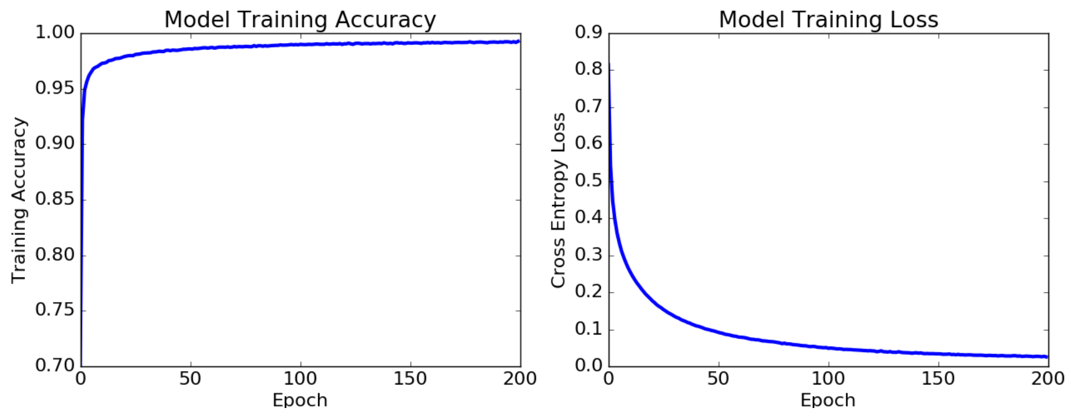


Figure 4.12. Training accuracy and training loss of SpectralNet trained on all training data.

Model training occurred on a GPU (GTX 1060, 6 GB video card RAM) and it took 3 seconds per epoch for 100 000 data augmented training spectra to be passed forward and backwards through the graph in batches of 2048. All neural network training was performed using Google's machine learning libraries Tensorflow and Keras in Python. A Github repository of the training scripts can be found at the following URL:

https://github.com/MichaelChatzidakis/Mn_Classifier_CNNs

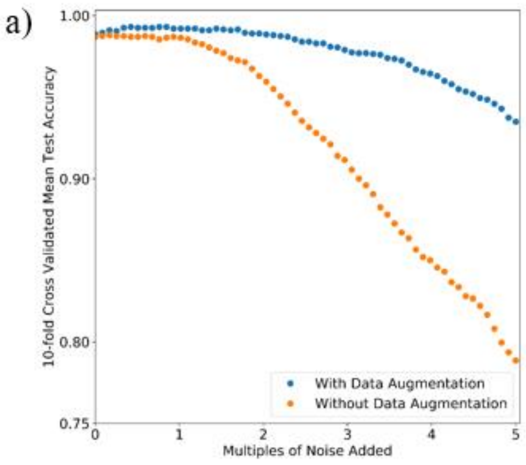
4.6.2 Model evaluation using cross-validation

To probe the generalizability of a convolutional neural network classifier in the context of spectroscopy, three accuracy metrics were used to benchmark how well the classifier generalizes to new data. These three accuracy metrics were used for model evaluation and are based on using domain knowledge from core-loss electron spectroscopy to better characterize the robustness of the proposed classifier. The first metric is a stratified 10-fold cross-validation validation-set accuracy, the second is the accuracy at 5x the base-level noise, and finally the accuracy when the validation-set is perturbed ± 5 eV (translation variance).

The overall validation-set accuracies were averaged over all 10 folds from cross-validation. The overall cross-validation test accuracy averaged over all folds was 99.85 % where the highest single evaluation was 100.0 % and the lowest was 98.5 %.

4.6.3 Noise invariance accuracy

To test the robustness of SpectralNet to the noise frequently found in transmission electron microscopes, a noise test was implemented using PCA. PCA was performed on each validation-set during cross-validation and low variance principal components were calculated on the validation-set and added back to the validation data in different scalar multiples. The aim of this test is two-fold. First, we tested whether PCA, used as a training data augmentation step, is effective by comparing the results with and without training data augmentation (Figure 4.13). Secondly, we also tested the classifier to probe how well SpectralNet can predict accurately the valence in the presence of extreme noise that a trained human spectroscopist would have great difficulty in classifying. The comparison between SpectralNet trained with or without data augmentation is shown in Figure 4.13a. As an example of what the signal-to-noise ratio looks qualitatively, the effect of different scalar multiples of low variance principal components on spectra is shown Figure 4.13b. This test shows that, in the presence of noise that is a 5x scalar multiple of low variance principal components, the data augmented classifier is able to exceed 93% validation-set accuracy. Without data augmentation the accuracy decreases moderately as noise is added with a validation-set accuracy of 78% at 5x the base-line noise level. It should be noted that a 1x scalar multiple of the low variance principal components is indeed just the original signal. A 0x scalar multiple in this context is a PCA cleaned spectra with low variance components being removed.



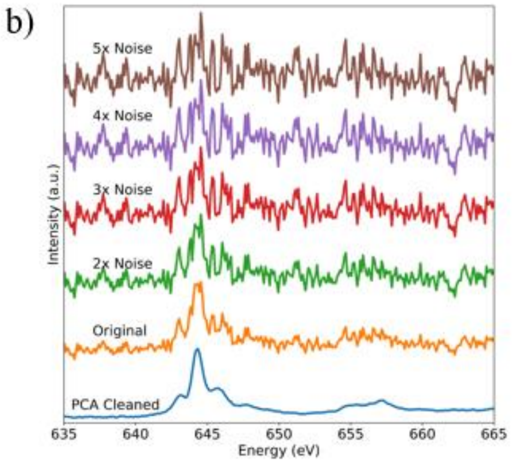


Figure 4.13. a) A comparison of the effect of data augmentation on how SpectralNet performs when adding multiples of low variance principal components. b) An example of spectra

showing the effect of scalar multiples of low variance principal components added on to spectra.

4.6.4 Translation invariance accuracy

The effect of calibration and chemical shift on classification accuracy was also explored as the third evaluation metric. Depending on the calibration of the spectrometer, it is possible for peaks to be shifted. Because of this, it is crucial that a classifier must understand the shape of the spectra and not the onset of the peak. Therefore, an ideal spectroscopy optimized neural network architecture would be one where there is no need to explicitly train the network using data augmentation to learn translation-invariance. Translation-invariance was measured by cropping the acquired spectra and moving the peaks into different positions into the 300 length input vector. A 5 eV shift in this context is equivalent to moving the peak 50 bins out of the total 300 bins. To test translation-invariance, the validation-set spectra for each fold were shifted left and right up to 5 eV. An example of this is shown in Figure 4.14b.

Translation-invariance is ideal in fields of spectroscopy where calibration is difficult or chemical effects may down- or up-shift the spectra and it is the shape of the spectra which is used for discrimination. However in other spectroscopies where peak shapes are identical, but peak position is the main discriminating feature (like XPS), translation invariance is not ideal. In this case, during model training using gradient descent, the weights of SpectralNet will simply not be tuned for translation invariance if that is not useful for discrimination. We observe this behavior here in EELS *because* of the type of classification problem we have. Neural networks, after all, are universal function approximators. SpectralNet will simply fit to the edge onset if that is the only single discriminating facet present in the data.

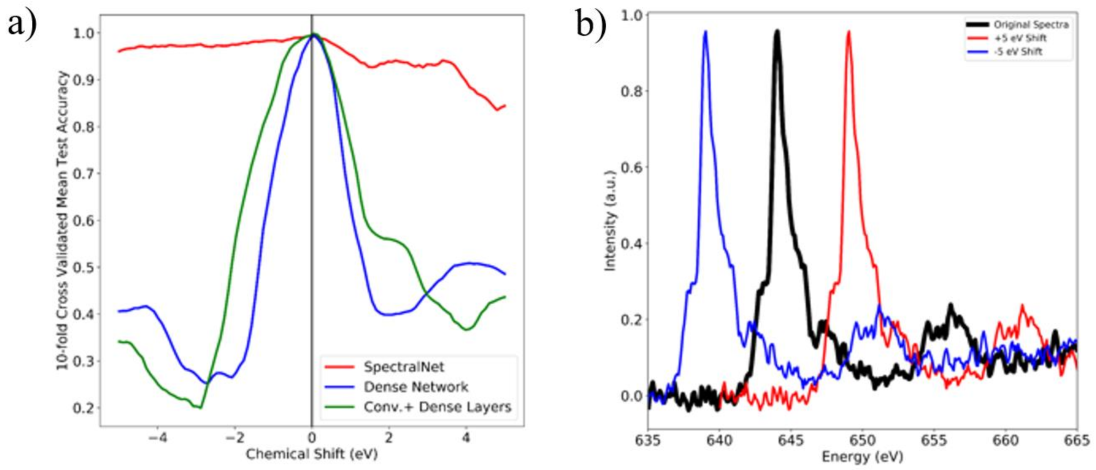


Figure 4.14. a) A comparison of how accuracy changes with respect to shifting the input spectra with a variety of different types of model architectures. b) An example of translational shifts on the 300-length input vector.

In Figure 4.14a two other neural network architectures are compared against the 650 parameter SpectralNet. These other architectures were inspired by what has been already done in the literature in various forms of spectroscopies such as in X-ray absorption spectroscopy and Raman spectroscopy. In the first case, the *Dense Network* is a multi-layer neural network with two layers containing 32 dense neurons each with a total of 11,000 free parameters. This network is activated using a ReLU transfer function, regulated using 50% dropout and normalized using batch-normalization. This output is then put through a 3 neuron dense layer with a softmax activation function to produce predicted class probabilities. The second comparison is done with the *Convolutional* + *Dense Layers* network (operated with 1100 parameters). This neural network has the same convolutional feature extractor as SpectralNet but its classification block consists of a multi-layer neural network with 8 neurons in the first layer, 4 neurons in the second and with the same dropout, batch-normalization and softmax parameters as the fully dense network.

SpectralNet is shown to produce a small decrease in accuracy as the spectra are translated in the input vector whereas the other two architectures have a sharp (<60%) decrease in accuracy. It should be noted that the decrease in accuracy after +3 eV shifts is due to some Mn^{4+} spectra being cropped out of the range of the input vector.

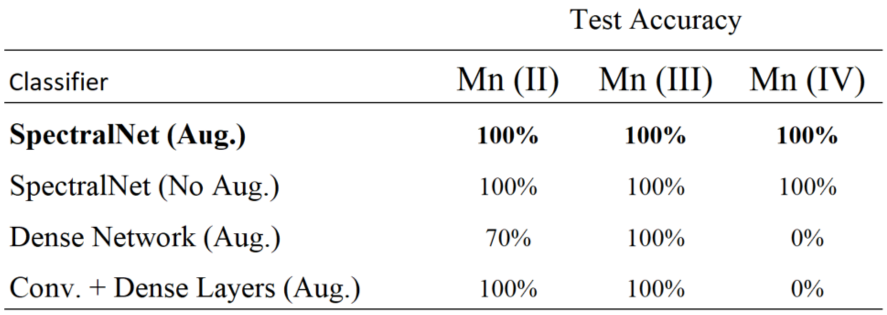
It is apparent that networks with dense layers fail to generalize when using test data that lie outside of the training distribution (3 eV chemical shifts maximum) whereas SpectralNet, a fully convolutional classifier, is immune to these perturbations. SpectralNet with its classification block containing depth-wise convolutions and global average pooling, needs to linearly combine the 12 activation map features to discriminate between classes. Whereas the dense classifiers require flattened (1D) inputs and are making a classification based off of per-bin values which leads to strong overfitting to the training distribution.

4.7 Model Testing using Digitized Reference Spectra

4.7.1 Accuracy on reference spectra digitized from publications

The architecture of SpectralNet chosen by using the model validation tests as benchmarks. To test how well SpectralNet generalizes to real world data, it was tested against 31 digitized spectra taken from the publications by Zhang *et al*., Tan *et al*, and Garvie *et al*. to probe generalizability in the presence of different instruments, calibration, and resolution. This dataset contains 12 Mn^{2+} spectra of various minerals (tetrahedral, octahedral, and dodecahedral coordination), 10 Mn^{3+} spectra of various minerals (octahedral), and 9 Mn^{4+} spectra of various minerals (octahedral). These spectra are also shifted significantly (~ 3 eV) compared to the acquired spectra, likely the result of different instrument calibration and have different levels of noise (Figure 4.2).

The classifiers shown in Figure 4.14 were retrained using all of the training data and tested against the 31 digitized reference spectra. The test accuracies on the digitized reference spectra dataset are shown in Figure 4.13.



	Test Accuracy		
Classifier	Mn (II)	Mn (III)	Mn (IV)
SpectralNet (Aug.)	**100%**	**100%**	**100%**
SpectralNet (No Aug.)	100%	100%	100%
Dense Network (Aug.)	70%	100%	0%
Conv. + Dense Layers (Aug.)	100%	100%	0%

Figure 4.15. The test accuracy for each valence of Mn taken from the digitized reference spectra. SpectralNet is compared with and without data augmentation. SpectralNet is also compared against a fully dense neural network with convolutional feature extractor and a dense classifier network to highlight the loss of generalizability when using dense layers.

SpectralNet is proven to be extremely successful on the digitized reference spectra dataset even without data augmentation. It was thought that data augmentation would be required for SpectralNet to understand such clean data since some of the augmented spectra would be PCA-cleaned training data, but it was proven to not matter.

SpectralNet is also capable of classifying Mn compounds that do not have the same coordination (octahedral) as the acquired training data such as tetrahedral and dodecahedral compounds. This is due to the fact that, as demonstrated by Garvie *et al* and Tan *et al*, there are similarities in the shape of the fine-structure of the core-loss edges between various Mn, Fe and V compounds of the same valence (Garvie & Craven, 1994; Tan *et al*., 2012) for different coordinations.

On the digitized reference spectra dataset, dense layer containing architectures fail to generalize to data outside of their training distributions. In particular these classifiers cannot find the differences between trivalent and tetravalent Mn because it is only a small shoulder that separates them. To understand why this may be the case, the following section will look into using visualization tools to understand what these networks are doing.

4.7.2 Feature visualization using t-distributed stochastic neighbor embedding

To help explain the success of SpectralNet on the digitized reference spectra dataset, the 300-length preprocessed spectra (both acquired and reference spectra) were projected onto a 2D plane using a t-distributed stochastic neighbor embedding (t-SNE) technique. t-SNE developed by van der Maaten and Hinton is a popular dimensionality reduction technique which attempts to preserve the distribution of clusters in the original high-dimensional space when projecting the data onto a 2D plane for visualization purposes (Van Der Maaten & Hinton, 2008). Note that the analysis presented here is entirely qualitative to aid in exploratory visualization. The visualizations shown in Figure 4.16 were initialized with PCA, have a perplexity of 100, and were run for 10 000 iterations. A variety of visualizations with perplexities between 20 and 100 can be found in the Appendix to prove convergence. The perplexity can be thought of as a similar metric as choosing the number of cluster centers when doing k-Means clustering. It is a description of the density of neighbors found in a Gaussian of fixed variance around the point in question. It is this nearest neighbor density that is attempted to be preserved when projected from the higher dimensional plane to the 2D plane.

The features produced by the final feature extraction layer of SpectralNet, consisting of 12 vectors of length 6, was flattened into a 72-length vector and also visualized using t-SNE. The classes can be easily differentiated on the feature space shown in Figure 4.16b. In the feature space, it is evident that the reference spectra (the triangles in Figure 4.16) are contained within the clusters of acquired spectra. This is in contrast to the t-SNE visualization of the input space where all Mn^{4+} are incorrectly classified as Mn^{3+}.

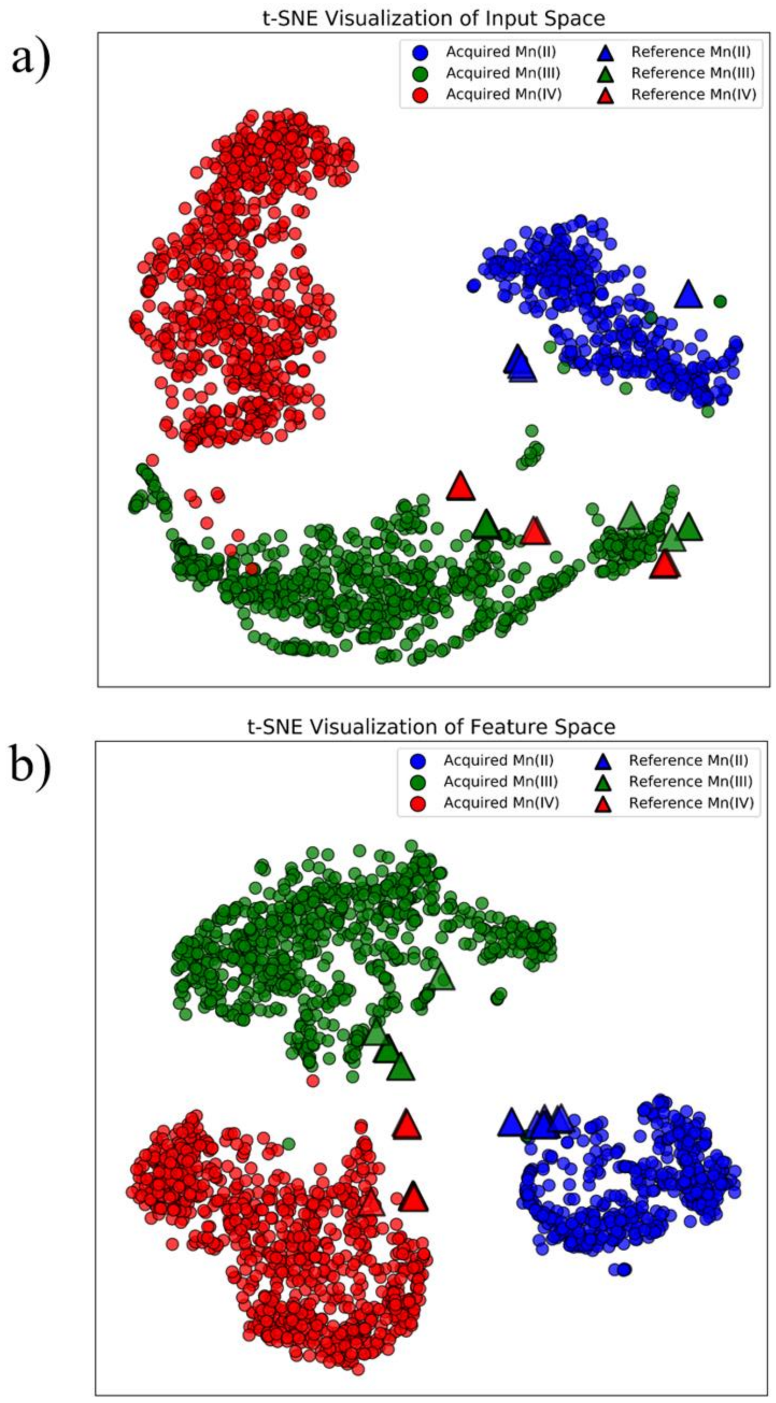


Figure 4.16. 2D t-SNE visualization of the 300-dimensional input space (a) and of the 72-dimensional feature space of the trained SpectralNet (b). The manifold of the feature space brings the digitized reference spectra (triangles) closer to the clusters of the acquired data as opposed to the original input space.

Figure 4.14 highlights that the feature representation created by SpectralNet is calibration invariant, noise invariant and atomic coordination invariant. If sharp decision boundaries were to be made on the original data (Figure 4.14a), like a dense neural network would do, then it is clear that some of triangles (test data) would be incorrectly classified as belonging to the wrong cluster. Even the feature representation domain shown in Figure 4.14b can be misleading and sharp boundaries drawn on the outskirts of the acquired data clusters would also not be ideal, which is why the *Convolutional + Dense Network* failed to generalize. SpectralNet is capable of drawing soft boundaries around the clusters because it has a very constrained classification architecture.

The digitized reference spectra dataset is very different from the acquired dataset (cleaner signal and shifted 3 eV). It is evident from the t-SNE visualization of the full input spectra space that these two datasets lie in different distributions. We can gauge this qualitatively by observing that reference Mn4+ are more closely similar to acquired Mn3+ than they are to acquired Mn4+.

A comparison of the intermediate activations of the digitized reference spectra for each vertex in SpectralNet's graph is can be found in the Appendix. These are directly compared to the activations shown in Figure 4.9.

4.8 Tolerance of SpectralNet to superposition of classes (hybrid valence case)

It is of interest to the Li-ion battery community to not only classify integer valence, but *hybrid valence* as well. By hybrid valence, we do not mean literally a fractional formal charge, but rather we refer to a superposition of multiple atoms of Mn with different oxidation states caught in the same spectrum during acquisition. The signals of each oxidation state will be overlapping and in proportion to how much was found in the sample. For instance this can be the case in a technique like x-ray photoelectron spectroscopy (XPS) or Mossbauer spectroscopy where the signal is captured over a large length scale (microns), and what is collected is the superposition (i.e. the sum) of all of the atoms of different oxidation states in that micrometer scale (or larger) region of space excited by the photon beam. In these bulk techniques, typically a least-squares fitting using reference standards is used to approximate the fractions of each valence.

The advantage of using the fine-structure of EELS to map valence as opposed to other techniques is the incredibly high spatial resolution. XPS typically can do valence mapping on the micron scale whereas as will be shown in the following section, SpectralNet is able to map valence at the angstrom scale using EELS. Such mapping is crucial in understanding how valence changes on the first few nanometers of a battery compound. XPS would only be able to provide a rough estimate with such limited spatial resolution.

4.8.1 Linear superposition training using data augmentation

SpectralNet in its simplest form cannot approximate mixed valence. It is tempting to want to use the predicted class probabilities to approximate a mixed signal, but this is naïve. The class probabilities are not to be understood as such because they represent the creation of a decision boundary on the **training data**. It is an estimate, assuming the incoming test data falls within the training distribution. Clearly a mixed valence system is well out of scope of what is encapsulated in the training distribution and the probabilities are not the probabilities we are looking for. Instead we must add additional classes to the classification problem, such as $Mn^{2.5+}$ and $Mn^{3.5+}$. Any intermediate values between valence states are fine, but in this study only those two were explored.

Hybrid valence spectra can be created using the data that is already acquired of the pure oxidation states. By taking a linear combination of a randomly selected spectra and adding them to other randomly selected spectra of a different valence we can produce an artificial mixed valence spectra. However there is one major problem with a technique like this since we must rely on relative calibrations and chemical shift between spectra. In the past, these translational shifts were not problematic because we were looking at isolated spectra. However when taking a sum of two spectra, depending on how shifted the individual valences are, the superposition will look substantially different. To get around this problem, data augmentation had to be used. Each randomly chosen spectra was also randomly shifted by ±1 eV. This is shown in Figure 4.17 and is responsible for the variability in signals for both $Mn^{2.5+}$ and $Mn^{3.5+}$.

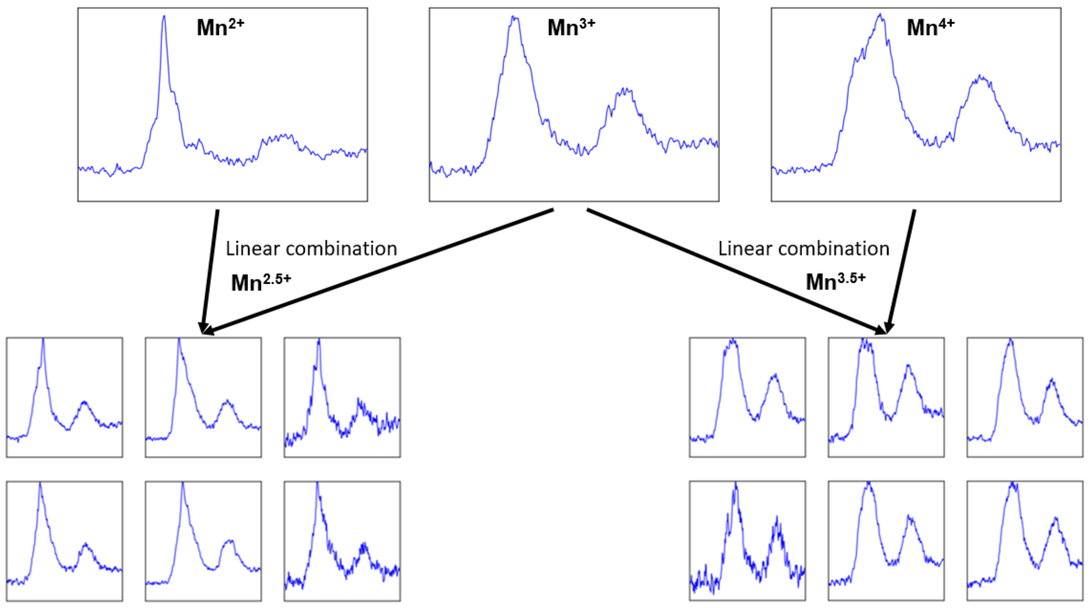


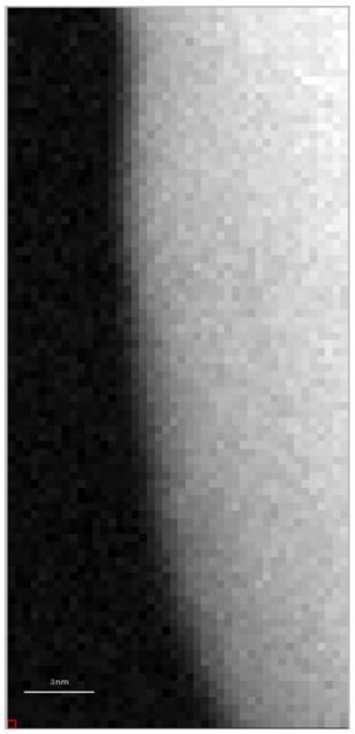
Figure 4.18. Schematic of how mixed valence spectra are created by linear combination of randomly selected spectra of each valence. In addition to doing a linear combination, the spectra are also randomly shifted ±1 eV to increase the variance.

SpectralNet was retrained using the same three pure oxidation states as three classes, and an additional 2 new classes for the mixed valences. 1500 spectra, each of mixed valence, were created to try and encapsulate a wide variety of shifts and mixtures. Each of the 5 classes were then data augmented by a factor of 25 using the PCA data augmentation scheme described before between 0 and 5 times the base-line noise. Drop-out was reduced from 0.85 to 0.1 to allow more features to be used for the discrimination task. Convergence occurred under 100 epochs and the retrained SpectralNet was able to reach 93 % accuracy on the original digitized reference spectra dataset.

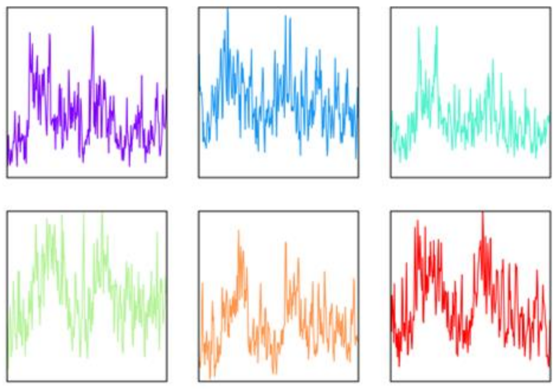
4.8.2 Hybrid valence atomic mapping using superposition-trained SpectralNet

To test how well SpectralNet is capable of handling mixed valence cases, a spectrum image of the surface of a grain of a nickel-manganese-cobalt (NMC) Li-ion cathode material was obtained from Hanshuo Liu, a graduate student who did a PhD degree on the study of NMC structure and degradation as a function of electrochemical cycling. This material is extremely beam sensitive and can only handle a very low dose of electrons. As such the spectra from this sample are very noisy and the material is difficult to image in its pristine state. The spectrum image is shown below in Figure 4.19a and 6 randomly selected spectra from 6 pixels of the spectrum image are shown in Figure 4.19b. These spectra qualitatively look very noisy but by performing PCA on the data it is apparent that the spectra can be cleaned with minimal loss of signal. The majority of the signal (~99%) can be explained with only three principal components as shown in the scree plot of Figure 4.19c. A description of scree plots can be found in the *Principal Component Analysis* section.

a)



b)



c)

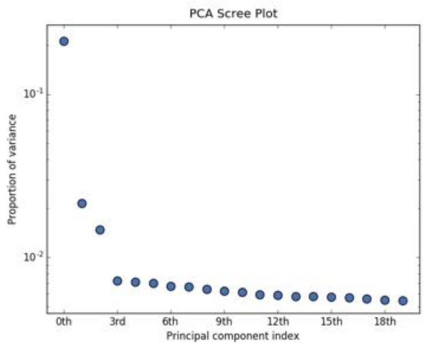


Figure 4.19. a) Spectrum image of an NMC cathode. b) Randomly selected representative single pixel spectra of the Mn EELS signal found in the spectrum image. C) Scree plot describing the principal components of the spectrum image.

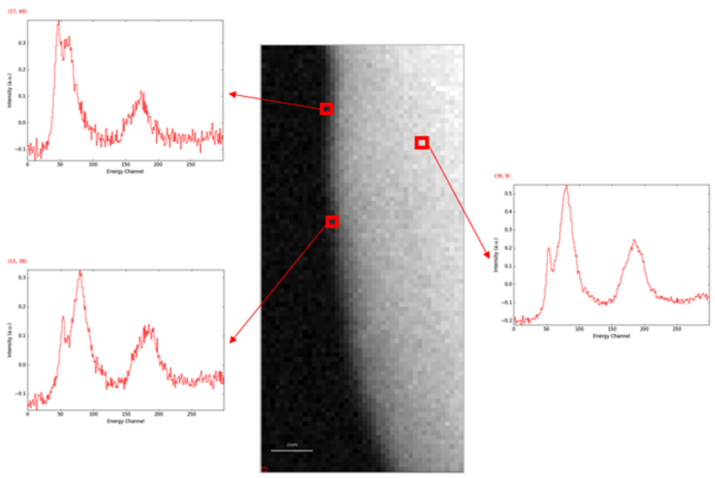
The spatial resolution (i.e. the pixel size) of the spectrum image is 3 Å indicating that each pixel is small enough for valence mapping at the nanoscale. The sample is still quite thick (> 50 nm) so it will not be single atom mapping since all counts through thickness will be collected.

While SpectralNet is capable of handling high noise, the noise present in this sample is quite large due to the need to use low dose conditions for data acquisition. PCA is needed to remove the low variance principal components to increase the signal-to-noise ratio. After denoising the spectrum image using PCA, some pixels are found to still be noisy but others are very clean as shown in Figure 4.20a. To threshold substrate pixels vs. NMC cathode particle pixels, k-Means clustering was used. Everything labelled as a substrate pixel was colored black. Some pixels within the sample are colored black indicating that the algorithm thought it was substrate due to the level of noise at that particular pixel.

To this denoised spectrum image, SpectralNet performed inference on every pixel and the resulting classification was mapped using a rainbow color scale as shown in Figure 4.20b. According to SpectralNet, a gradient of Mn oxidation states is observed. This gradient only exists in the first 1 – 2 nm into the cathode material. This sort of spatial resolution is simply not possible with any other method besides EELS. There is a mixture of divalent and trivalent Mn at the surface, before transitioning into pure trivalent Mn, then into pure tetravalent Mn. The surface appears to be heterogeneous with some Mn^{4+} pixels and $Mn^{2.5+}$ pixels occurring on the surface. This can be observed when comparing the two highlighted surface pixels in Figure 4.20a. SpectralNet was able to also map some of the surface atoms as Mn^{4+} despite the majority of them being $Mn^{2.5+}$. These pure tetravalent Mn atoms may be reconstructed or they may be adsorbed onto the surface of NMC particles since it appears that they only exist on the outermost monolayer.

While this analysis could be performed by using PCA as a data cleaning step, then by using a least-squares fitting of reference compounds, the advantage of SpectralNet is that one does not need to decide one candidate of reference compound. As was described in the previous section about the superposition of inputs, depending on how translated each input spectra is, the results will be incredibly different. There will be an extensive amount of manual tweaking to get a least-squares fitting to work well. SpectralNet however has no such ambiguity and once it is trained it is applicable to all mixed valence cases.

a)



b)

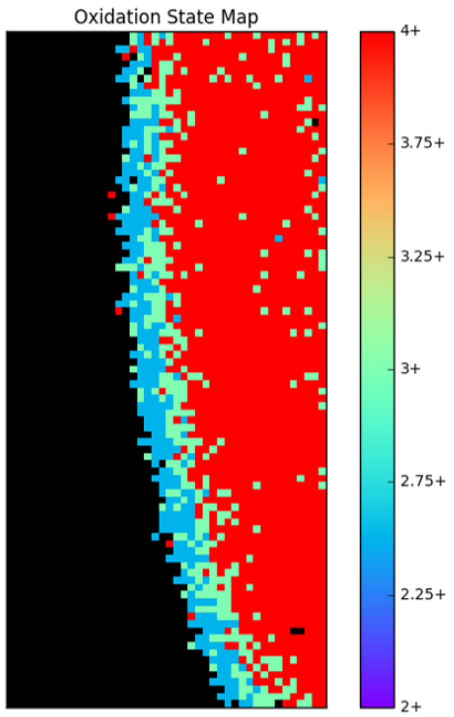


Figure 4.20. a) Three spectra selected from different locations on the sample after being cleaned via PCA. b) Oxidation state mapping using SpectralNet for hybrid valences.

5 Conclusion

In the field of electron energy loss spectroscopy (EELS), spectrometer calibration is often very difficult and leads to non-reproducible data between research groups. In addition to this, the noise profile is very different between instruments so typically dimensionality reduction (unsupervised learning methods or manual feature extraction) of the data needs to be performed to compare results between instruments. In this study, we create a new method of spectroscopy analysis using deep convolutional neural networks. These networks are proven to have significant advantages over all other methods in EELS analysis. Our neural network is immune to calibration differences and have high noise tolerances which exceeds the ability of the currently used methods.

Convolutional neural networks are automated feature extractors. They require no domain knowledge, and require no manual feature collection, such as taking the FWHM of peaks or determining peak on-set. It is clear from the t-SNE visualization (Figure 4.16) that the feature “fingerprints” produced by SpectralNet successfully embed discriminating differences between the three valence clusters from both reference data digitized from publications and spectra that we acquired. The digitized reference spectra dataset is very different from the acquired dataset (cleaner signal and shifted 3 eV) and it is evident from the t-SNE visualization of the full input spectra space that these two datasets lie in different distributions. Despite this, SpectralNet is proven to successfully perform classification.

The architecture of SpectralNet was created using three accuracy metrics as a benchmark: stratified 10-fold cross-validation validation-set accuracy, accuracy at 5x the baseline noise level, and accuracy with large translational shifts of peaks. SpectralNet has an average validation-set accuracy of 99.85% and a 5x noise accuracy of 93%. When translating the peaks of spectra along the energy axis (which can be considered the time axis) by large (±5 eV) shifts, SpectralNet still retains high accuracy and it is evident that this network has developed a translation-invariance without being explicitly trained on data augmented examples of translational shifted spectra. In contrast, the accuracy during translational shifting of a dense-layer containing neural network is consistently lower, however, and similar to that of a random guess of 33% at large translational shifts.

In conclusion, the success of the generalizability of SpectralNet in the classification of high noise and/or non-calibrated time-series (i.e. spectroscopy) data relates to its triangular feature extractor and simple classification architecture. Using gradient descent to solve for the set of parameters that will maximally discriminate the classes in question, allows the triangular feature extractor to be optimized for finding the differences between the spectra. These differences are magnified as the input spectra are propagated through the network, and the input spectra become transformed into a set of features which are a lower-dimensional fingerprint optimized for discrimination. This fingerprint approach is shown to be an excellent embedding of the spectra whereby a simple linear combination of features, and logistic regression, are sufficient for the classification of the unknown spectra in the presence of extreme noise or translational shifts. In fact, more complicated classification layers, such as a commonly used dense neural network layer, lead to a significant decrease in generalizability and strong overfitting to the training distribution. We demonstrate that dense hidden layers, with a 1D convolutional neural network feature extractor for a time-series classification task, lead to a decrease in generalizability with respect to temporal invariance in cases where the training data does not encapsulate enough temporal (here, energy) variance. This will be particularly relevant when training the classifier using one data set acquired with one spectrometer but testing it on another instrument with different calibration or on different sessions where stability of the instrument is not perfectly controlled. Therefore, classifiers like SpectralNet have the potential to remove qualitative methodologies from spectroscopy interpretation and can be extended to additional classes (valences, elements, bonding states etc.) provided there is training data. These classifiers can be applied to many other spectroscopic methods where discrimination between spectra is determined by peak shape, instances with high noise or cases where calibration is difficult and unreliable over time.

6 References

Bell, D., & Natasha, E. (2013). Low Voltage Electron Microscopy Principles and Applications. *Wiley*,

Bengio, Y. (2012). Practical recommendations for gradient-based training of deep architectures. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *7700 LECTU*, 437–478. http://doi.org/10.1007/978-3-642-35289-8-26

Bonnet, N., & Nuzillard, D. (2005). Independent component analysis: A new possibility for analysing series of electron energy loss spectra. *Ultramicroscopy*, *102*(4), 327–337. http://doi.org/10.1016/j.ultramic.2004.11.003

Bosman, M., Keast, V. J., García-Muñoz, J. L., D'Alfonso, A. J., Findlay, S. D., & Allen, L. J. (2007). Two-dimensional mapping of chemical information at atomic resolution. *Physical Review Letters*, *99*(8), 1–4. http://doi.org/10.1103/PhysRevLett.99.086102

Bosman, M., Keast, V. J., Watanabe, M., Maaroof, A. I., & Cortie, M. B. (2007). Mapping surface plasmons at the nanometre scale with an electron beam. *Nanotechnology*, *18*(16). http://doi.org/10.1088/0957-4484/18/16/165505

Bosman, M., Watanabe, M., Alexander, D. T. L., & Keast, V. J. (2006). Mapping chemical and bonding information using multivariate analysis of electron energy-loss spectrum images. *Ultramicroscopy*, *106*(11–12 SPEC. ISS.), 1024–1032. http://doi.org/10.1016/j.ultramic.2006.04.016

Carey, C., Boucher, T., Mahadevan, S., Bartholomew, P., & Dyar, M. D. (2015). Machine learning tools formineral recognition and classification from Raman spectroscopy. *Journal of Raman Spectroscopy*, *46*(10), 894–903. http://doi.org/10.1002/jrs.4757

Cueva, P., Hovden, R., Mundy, J. A., Xin, H. L., & Muller, D. A. (2012). Data processing for atomic resolution electron energy loss spectroscopy. *Microscopy and Microanalysis*, *18*(4), 667–675. http://doi.org/10.1017/S1431927612000244

Garvie, L. A. J., & Craven, A. J. (1994). High-resolution parallel electron energy-loss spectroscopy of Mn L2,3-edges in inorganic manganese compounds. *Physics and Chemistry of Minerals*, *21*(4), 191–206. http://doi.org/10.1007/BF00202132

Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural Networks*, *4*(2), 251–257. http://doi.org/10.1016/0893-6080(91)90009-T

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., … Adam, H. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. http://doi.org/arXiv:1704.04861

Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., & Keutzer, K. (2016). SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size, 1–13. http://doi.org/10.1007/978-3-319-24553-9

Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. http://doi.org/10.1007/s13398-014-0173-7.2

Keast, V. J., Scott, A. J., Brydson, R., Williams, D. B., & Bruley, J. (2001). Electron energy-loss near-edge structure - A tool for the investigation of electronic structure on the nanometre scale. *Journal of Microscopy*, *203*(2), 135–175. http://doi.org/10.1046/j.1365-2818.2001.00898.x

Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization, 1–15. http://doi.org/http://doi.acm.org.ezproxy.lib.ucf.edu/10.1145/1830483.1830503

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Advances In Neural Information Processing Systems*, 1–9. http://doi.org/http://dx.doi.org/10.1016/j.protcy.2014.09.007

Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. http://doi.org/10.1038/nature14539

Liu, J., Osadchy, M., Ashton, L., Foster, M., Solomon, C. J., & Gibson, S. J. (2017). Deep Convolutional Neural Networks for Raman Spectrum Recognition: A Unified Solution, 4067–4074. http://doi.org/10.1039/c7an01371j

Lopez-Reyes, G., Sobron, P., Lefebvre, C., & Rull, F. (2014). Multivariate analysis of Raman spectra for the identification of sulfates: Implications for ExoMars. *American Mineralogist*, *99*(8–9), 1570–1579. http://doi.org/10.2138/am.2014.4724

M., G., & P., D. (2002). Neural networks and the classification of mineralogical samples using x-ray spectra. *Proceedings of the 9th International Conference on Neural Information Processing.*, *5*, 2683–2687. http://doi.org/10.1109/ICONIP.2002.1201983

Maigné, A., & Twesten, R. D. (2009). Review of recent advances in spectrum imaging and its extension to reciprocal space. *Journal of Electron Microscopy*, *58*(3), 99–109. http://doi.org/10.1093/jmicro/dfp022

Potapov, P. L., & Schryvers, D. (2004). Measuring the absolute position of EELS ionisation edges in a TEM. *Ultramicroscopy*, *99*(1), 73–85. http://doi.org/10.1016/S0304-3991(03)00185-2

Radtke, G., & Botton, G. A. (2011). *Scanning Transmission Electron Microscopy*. http://doi.org/10.1007/978-1-4419-7200-2

Roth, J. A., Kaeberle, M. L., & Hsu, W. H. (1982). Effect of estradiol and progesterone on lymphocyte and neutrophil functions in steers. *Infection and Immunity*, *35*(3), 997–1002. http://doi.org/10.1002/jemt

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*(6088), 533–536. http://doi.org/10.1038/323533a0

Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition, 1–14. http://doi.org/10.1016/j.infsof.2008.09.005

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, *15*, 1929–1958. http://doi.org/10.1214/12-AOS1000

Tan, H., Verbeeck, J., Abakumov, A., & Van Tendeloo, G. (2012). Oxidation state and chemical shift investigation in transition metal oxides by EELS. *Ultramicroscopy*, *116*, 24–33. http://doi.org/10.1016/j.ultramic.2012.03.002

Timoshenko, J., Lu, D., Lin, Y., & Frenkel, A. I. (2017). Supervised Machine-Learning-Based Determination of Three-Dimensional Structure of Metallic Nanoparticles. *Journal of Physical Chemistry Letters*, *8*(20), 5091–5098. http://doi.org/10.1021/acs.jpclett.7b02364

Van Der Maaten, L. J. P., & Hinton, G. E. (2008). Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, *9*, 2579–2605. http://doi.org/10.1007/s10479-011-0841-3

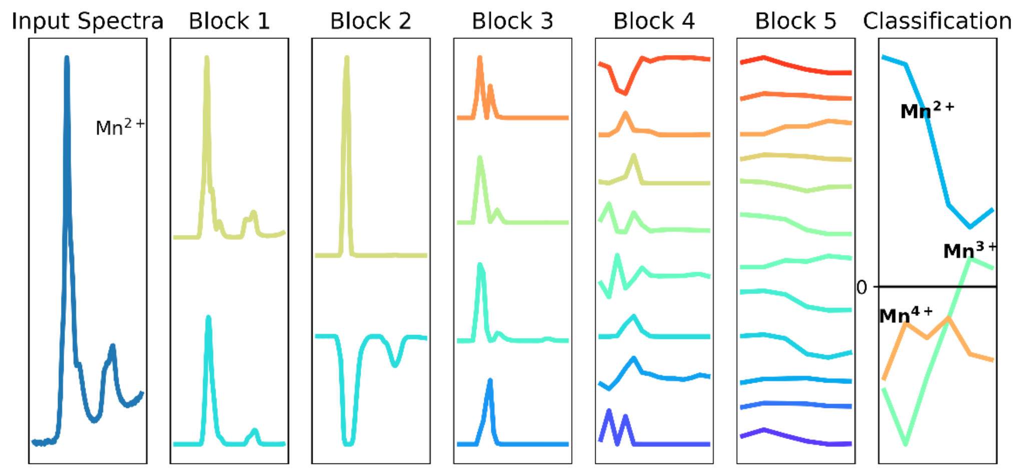
Yedra, L., Xuriguera, E., Estrader, M., López-Ortega, A., Baró, M. D., Nogués, J., … Peiró, F. (2014). Oxide wizard: An EELS application to characterize the white lines of transition metal edges. *Microscopy and Microanalysis*, *20*(3), 698–705. http://doi.org/10.1017/S1431927614000440

Zhang, S., Livi, K. J. T., Gaillot, A.-C., Stone, A. T., & Veblen, D. R. (2010). Determination of manganese valence states in (Mn3+, Mn4+) minerals by electron energy-loss spectroscopy. *American Mineralogist*, *95*(11–12), 1741–1746. http://doi.org/10.2138/am.2010.3468

Zheng, C., Mathew, K., Chen, C., Chen, Y., Tang, H., Dozier, A., … Ong, S. P. (2017). Automated Generation and Ensemble-Learned Matching of X-ray Absorption Spectra. *ArXiv E-Prints*. Retrieved from http://arxiv.org/abs/1711.02227

7 Appendix

7.1 Supporting Figures



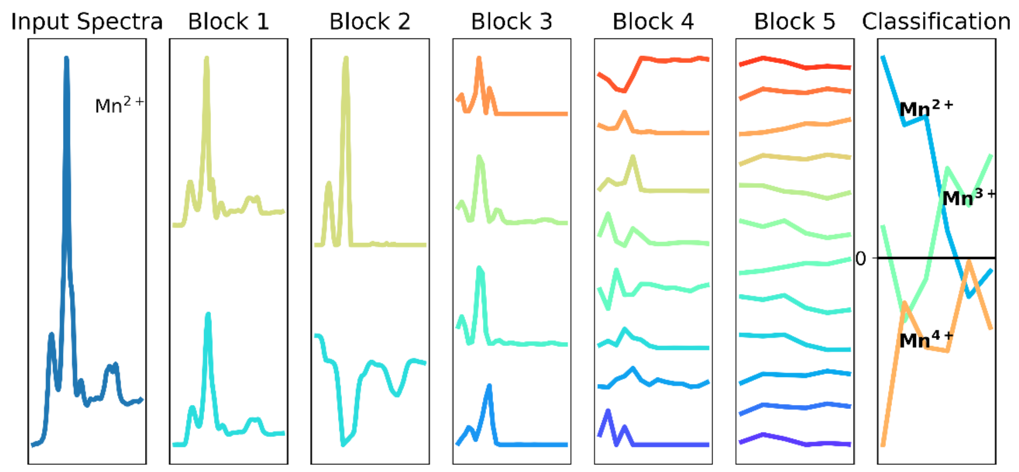
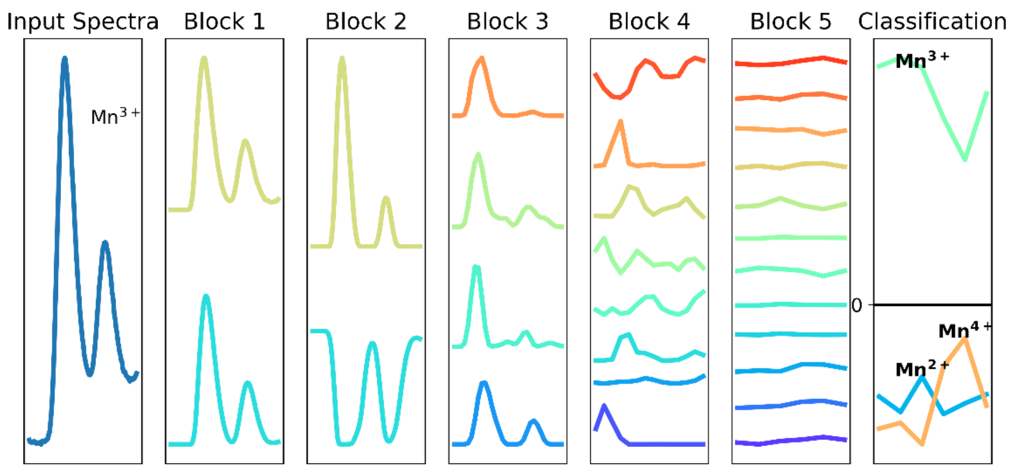


Figure S1. (top) Intermediate activations averaged over all acquired Mn^{2+} spectra. (bottom) Intermediate activations averaged over all digitized reference Mn^{2+} spectra.



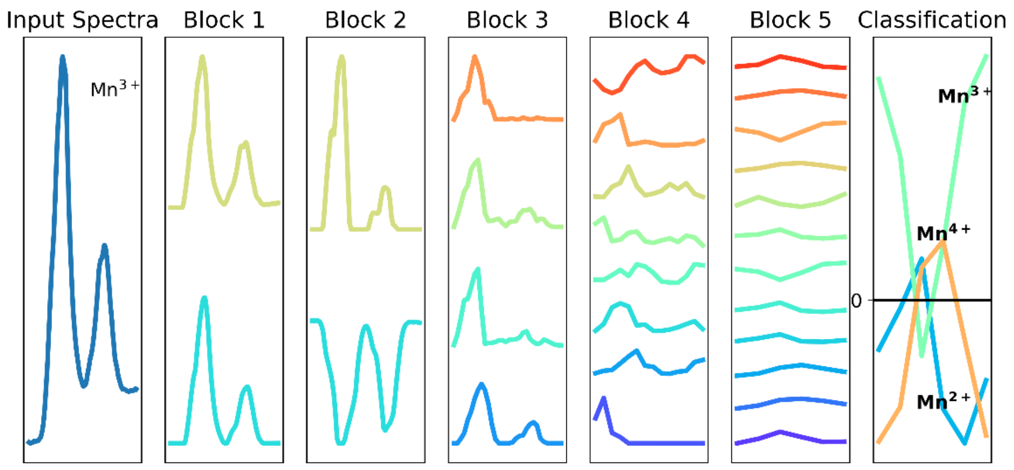
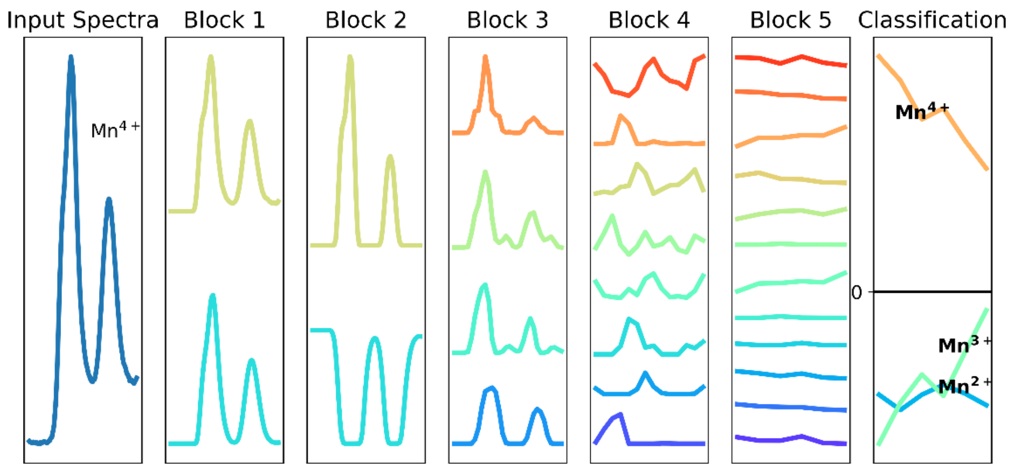


Figure S2. (top) Intermediate activations averaged over all acquired Mn^{3+} spectra. (bottom) Intermediate activations averaged over all digitized reference Mn^{3+} spectra.



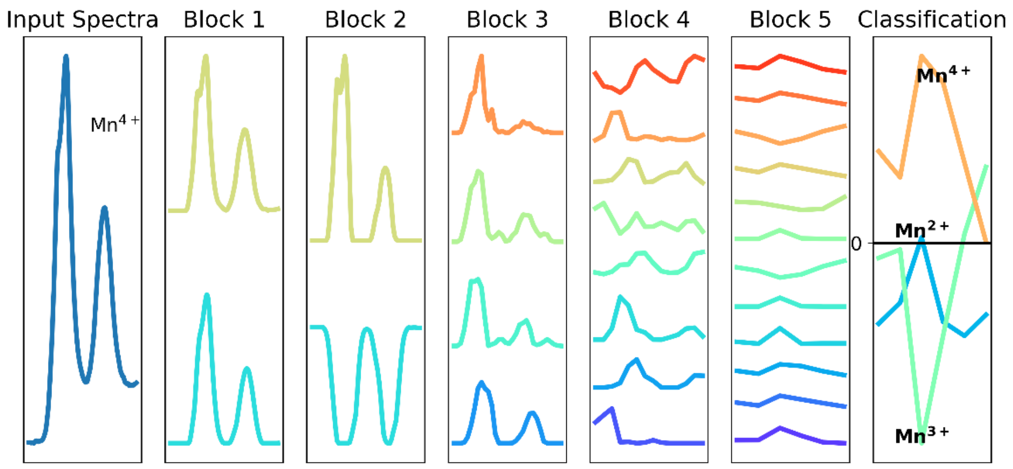


Figure S3. (top) Intermediate activations averaged over all acquired Mn^{4+} spectra. (bottom) Intermediate activations averaged over all digitized reference Mn^{4+} spectra.

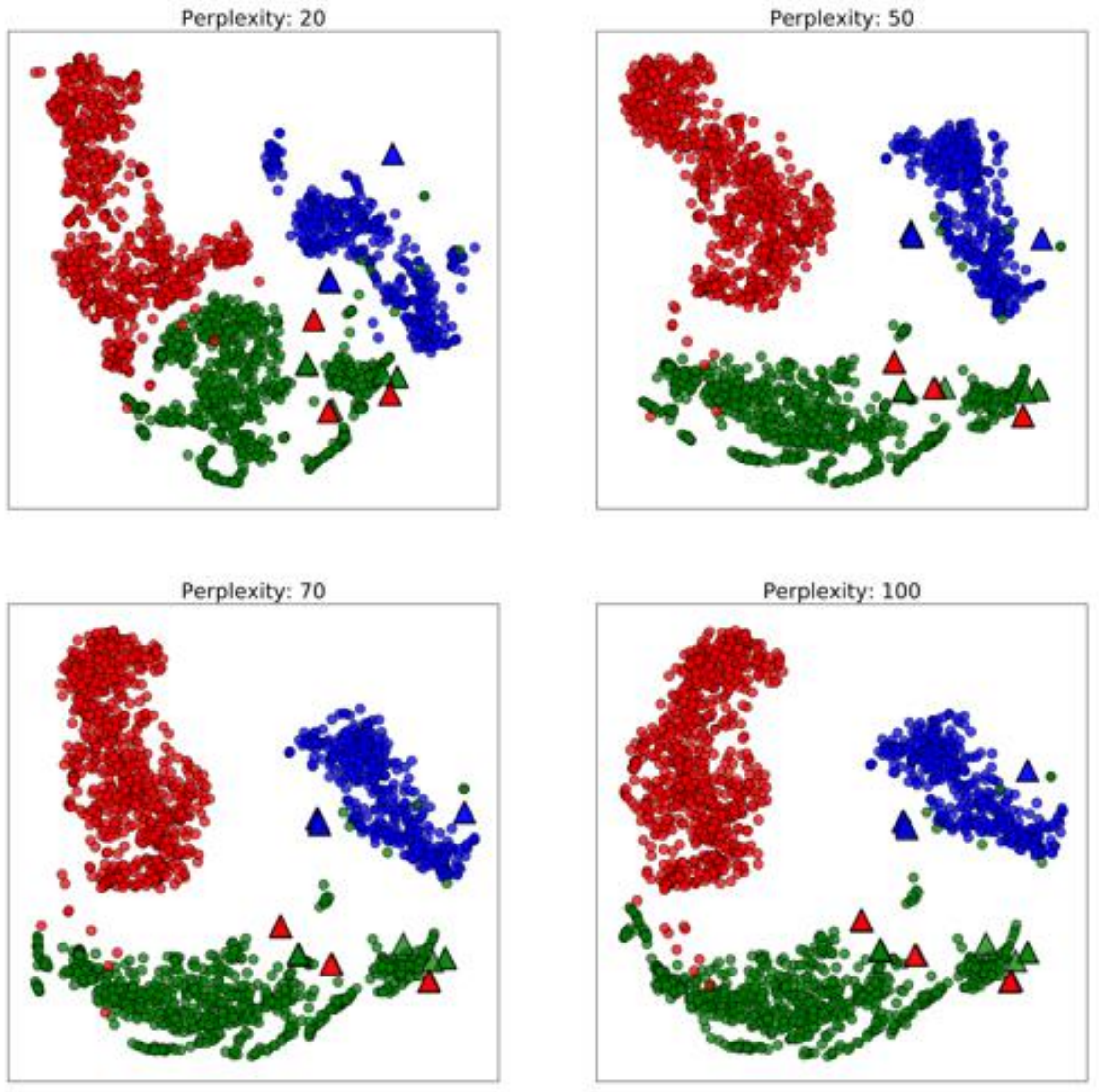


Figure S4. t-SNE of the input spectra space across many values of perplexity. Circles represent acquired data whereas triangles are digitized reference spectra. Blue, green and red represent Mn^{2+}, Mn^{3+}, and Mn^{4+} respectively.

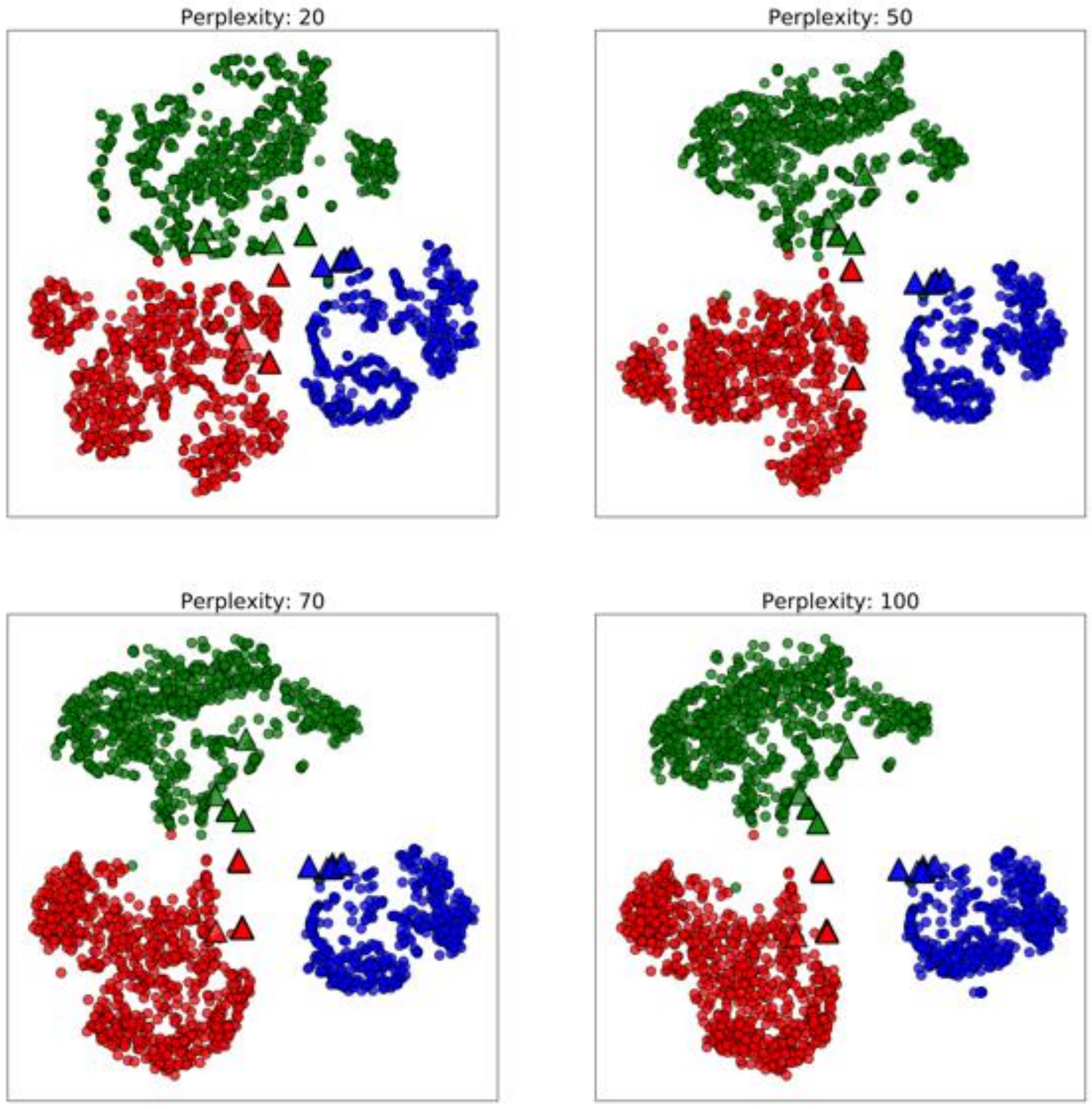


Figure S5. t-SNE of the feature space across many values of perplexity. Circles represent acquired data whereas triangles are digitized reference spectra. Blue, green and red represent Mn^{2+}, Mn^{3+}, and Mn^{4+} respectively.