

**Explainable Predictive Modeling  
and Synthetic Sample Generation  
for Limited Spectral Data**

BY

FRANTISHEK AKULICH

B.S., University of Illinois at Chicago, 2020

THESIS

Submitted as partial fulfillment of the requirements  
for the degree of Master of Science in Industrial Engineering  
in the Graduate College of the  
University of Illinois at Chicago, 2022

Chicago, Illinois

Defense Committee:

Dr. Hadis Anahideh, Chair and Advisor

Dr. Kenneth Brezinsky

Dr. Patrick Lynch

Dr. Houshang Darabi

## Acknowledgements

I would like to express a deep appreciation to my supervising professor Dr. Hadis Anahideh for her guidance and patience throughout this program, as well as the wealth of knowledge about the area of Data Science that I inherited from her. I am also thankful to Dr. Jonathan Komperda and Dr. Quintin Williams for their unwavering support, belief, and encouragement during the years at UIC. I would like to thank Dr. Kenneth Brezinsky and Dr. Patrick Lynch for providing me such an extraordinary and exciting opportunity to work on this project and for always listening and providing kind and honest advice. I would I want to extend my appreciation to my kind committee member Dr. Houshang Darabi, as he was the first Professor at UIC who inspired me to learn about Machine Learning. Finally, I extend my appreciation to all the brilliant people at UIC, as well as friends and family outside of the University, that made this journey possible.

## **Contribution of Authors Statement**

Chapter §2 represents sections of published manuscript (1) for which I was the primary author contributing to Data curation, Methodology, Software, Investigation, Validation, Formal Analysis, Visualization, Writing - Original Draft, and Writing - Review and Editing. My supervisor Dr. Hadis Anahideh was responsible for: Conceptualization, Methodology, Investigation, Validation, Formal analysis, Writing - Original Draft, Writing - Review and Editing, Supervision, and Funding acquisition. Manaf Shayyeb assisted us with: Data curation, Software, Validation, and Formal Analysis. Dhananjay Ambre was responsible for: Data curation, Software, Validation, and Formal Analysis.

---

# Table of Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Feature Selection</b>	<b>9</b>
2.1	Background . . . . .	9
2.1.1	Related Works . . . . .	13
2.2	Methodologies . . . . .	18
2.2.1	Predictive Modeling . . . . .	18
2.2.2	Decomposition-based Feature Selection Techniques . . . . .	23
2.2.3	Model-based Feature Selection Techniques . . . . .	26
2.2.4	Model-agnostic Feature Selection Techniques . . . . .	27
2.2.5	Evaluation Metrics . . . . .	31
2.3	Experimental Results . . . . .	33
2.3.1	Domain . . . . .	33
2.3.2	Dataset . . . . .	35
2.3.3	Predictive Modeling For Spectra Data . . . . .	36
2.3.4	Feature Selection for Spectra Data . . . . .	43
2.3.5	Discussion . . . . .	49

<b>3 Synthetic Data Generation</b>	<b>58</b>
3.1 Background . . . . .	58
3.1.1 Related Works . . . . .	60
3.2 Methodologies . . . . .	66
3.2.1 Generative Adversarial Network . . . . .	66
3.2.2 Advanced GAN Architectures . . . . .	68
3.2.3 GAN Evaluation . . . . .	74
3.2.4 Data . . . . .	79
3.3 Experimental Results . . . . .	79
<b>4 Conclusion</b>	<b>99</b>
<b>Appendix A Elsevier License Agreement</b>	<b>103</b>
<b>References</b>	<b>108</b>

---

## List of Tables

2.1	Summary of feature selection methods, their advantages and disadvantages. . . . .	11
2.2	Summary of related work in comparison to this work . . . . .	16
2.3	SVR hyperparameters considered for optimization . . . . .	37
2.4	Keras Neural Network hyperparameters considered for optimization	37
2.5	SVR Complexity . . . . .	40
2.6	NN architecture generalization ability on Real-time data, recorded as Test MSE using different architectures on Full and Reduced sets of data	43
2.7	Performance of NN and SVR predictive models fitted with Full and Reduced sets of features. . . . .	57
3.1	ML Ablation test for Advanced GAN components . . . . .	83
3.2	Remaining concentration prediction result for first 10 mixtures . . . .	94

---

## List of Figures

2.1	Average (a) Training, (b) Validation and (c) Testing error of Shallow NN on reduced subsets for various number of trainable parameters in three scenarios . . . . .	44
2.2	Shallow NN vs DNN testing error on reduced subsets in Real-time scenario. . . . .	45
2.3	Average computation time of (a) SVR (b) NN on reduced subsets of data for Real-time scenario with various model complexity levels. . .	46
2.4	PCA cumulative sum of explained variance by number of components	47
2.5	PLS Regression cross-decomposition and prediction results by number of components . . . . .	47
2.6	Regions of importance selected by various techniques. (a) PCA, (b) PLS, (c) Random Forest, (d) Ridge, (e) SHAPley, (f) Global Surrogate, (g) LIME are highlighted with <i>red</i> and expert features are highlighted with <i>yellow</i> . . . . .	50
2.7	Method Correctness, displayed as proportion of wavenumbers matched with the domain-defined features. . . . .	52
2.8	Testing error for Fine-Tuned SVR and NN models on Full and Reduced subsets in Real-time scenario. . . . .	53

2.9	Testing Error on Real-time dataset vs Correctness trade-off for 120 selected features using (a) SVR and (b) NN models . . . . .	56
3.1	Complete synthetic data generation and evaluation process diagram.	75
3.2	Comparison between (a) real Raman training data and synthetic data generated using trained (b) Vanilla GAN model (c) GAN with Mode Specific (MS) normalization . . . . .	81
3.3	PCA first and second principal component overlay between real training spectra and synthetic spectra generated via GAN with mode specific normalization . . . . .	82
3.4	CN Prediction and synthetic data Labeling accuracy measured for both SVR and NN models. Four combinations of Oracle-Prediction models tested on Real-time data, (a) Oracle and (b) Prediction model evaluation . . . . .	85
3.5	Optimal GAN training loss history. . . . .	88
3.6	GAN Spectra generator optimized architecture. . . . .	90
3.7	CGAN Spectra critic optimized architecture. . . . .	91
3.8	The KS score for each wavenumber for CGAN output. . . . .	92
3.9	Pure alkane Mixtures ATR comparison between Real (red) and generated (black) data with 95% Confidence Interval (blue) across generating 10,000 samples of given spectra. CH <sub>2</sub> -CH <sub>3</sub> concentration (a) Alk 1: [0.848, 0.152] (b) Alk 2: [0,1] . . . . .	95
3.9	Pure alkane Mixtures ATR comparison between Real (red) and generated (black) data with 95% Confidence Interval (blue) across generating 10,000 samples of given spectra. CH <sub>2</sub> -CH <sub>3</sub> concentration (c) Alk 3: [0.7, 0.3] (d) Alk 4: [0.867, 0.133] . . . . .	96



3.9	Pure alkane Mixtures ATR comparison between Real (red) and generated (black) data with 95% Confidence Interval (blue) across generating 10,000 samples of given spectra. CH <sub>2</sub> -CH <sub>3</sub> concentration (e) Alk 5: [0.823, 0.177] (f) Alk 6: [0.808, 0.192] . . . . .	97
3.9	Pure alkane Mixtures ATR comparison between Real (red) and generated (black) data with 95% Confidence Interval (blue) across generating 10,000 samples of given spectra. CH <sub>2</sub> -CH <sub>3</sub> concentration (g) Alk 7: [0.789, 0.211] . . . . .	98

---

## List of Abbreviations

<b>ATR</b>	Attenuated Total Reflectance
<b>BO</b>	Bayesian Optimization
<b>CDF</b>	Cumulative Distribution Function
<b>CFG</b>	Chemical Functional Groups
<b>CGAN</b>	Conditional Generative Adversarial Network
<b>CH</b>	Carbohydrate
<b>CN</b>	Cetane Number
<b>CNN</b>	Convolutional Neural Network
<b>DCGAN</b>	Deep Convolutional Generative Adversarial Network
<b>DCN</b>	Derived Cetane Number
<b>EV</b>	Explained Variance
<b>FAME</b>	Fatty Acid Methyl Esters
<b>FTIR</b>	Fourier-transform Infrared Spectroscopy
<b>GAN</b>	Generative Adversarial Network
<b>GMM</b>	Gaussian Mixture Model
<b>GS</b>	Global Surrogate
<b>KS</b>	Kolmogorov-Smirnov Test
<b>LIME</b>	Local Interpretable Model-Agnostic Explanations

<b>LR</b>	Linear Regression
<b>MIR</b>	Mid-Infrared Spectroscopy
<b>ML</b>	Machine Learning
<b>MSE</b>	Mean Squared Error
<b>NIR</b>	Near Infrared Spectroscopy
<b>NMR</b>	Nuclear Magnetic Resonance Spectroscopy
<b>NN</b>	Neural Network
<b>PC</b>	Principal Component
<b>PCA</b>	Principal Component Analysis
<b>PLS</b>	Partial Least Squares
<b>ReLU</b>	Rectified Linear Activation Unit
<b>RF</b>	Random Forest
<b>RSS</b>	Residual Sum of Squares
<b>SHAP</b>	Shapley Additive Explanation
<b>SVM</b>	Support Vector Machine
<b>SVR</b>	Support Vector Regression
<b>WGAN</b>	Wasserstein Generative Adversarial Network

## Summary

To assess and optimize the performance of combustion systems, it is necessary to characterize fuel ignition quality. Acquiring ignition properties of fuels is a tedious process that involves sample preparation and ignition quality testing. With advances in alternate approaches to defining fuels, particularly through a digital spectroscopic signature, a new possibility to simplify and automate the process of getting such knowledge has emerged. In this work, we harness automated statistical learning to map between the underlying chemical properties of fuel and the spectroscopic data it is imprinted on. Ensuring that such mapping is accurate and interpretable, it establishes a pathway for efficiently and totally automating the extraction of various attributes from any fuel. However, the high-dimensional nature of spectroscopic data, its scarcity, and the noise associated with the data collection process are key roadblocks to accomplish such task.

In this research, we address these issues by integrating machine learning predictive modeling, interpretable feature selection techniques, and synthetic data generation, respectively. In the first part, we investigate the most commonly used feature selection techniques and adopt the most recent and advanced explainable AI techniques to interpret the prediction outcomes of high-dimensional and limited spectral data. Interpretation of the prediction outcome is beneficial for the domain experts as it ensures the transparency and faithfulness of the ML models to the domain knowledge. Due to the instrument resolution limitations, pinpointing important regions of the spectroscopic data creates a pathway to optimize the data collection process through the spectrometer device miniaturization. Reducing the device size and power, and hence, cost is essential for a real-world deployment of such a end-to-end system.

Furthermore, we consider a wide range of machine learning models that have been proven to be successful for the prediction of the Cetane Number of fuels. We specifically design three different scenarios to ensure that the evaluation of ML models is robust for the real-time practice of the developed methodologies and to uncover the hidden effect of various noise sources (statistical and from data collection) on the final outcome. The evaluation is performed for both the full model and reduced models using different feature selection techniques on a real dataset. In the second part, we devise a deep generative technique to produce high fidelity and high diversity synthetic spectroscopy samples learned from the original dataset to expand our limited data pool and improve the representation. Our developed GAN model is then evaluated using statistical similarity, prediction model efficacy, and domain-expert conformance metrics. The results indicate tangible improvement in prediction model generalization ability for unforeseen data. To further enhance the transparency of the entire process, we employ GAN to produce samples of a specific group of pure alkane mixtures and compare them to the expected output. We demonstrate that our data synthesis approach can learn and reproduce spectroscopic samples that have the physical attributes of real fuels despite being artificial.

# Introduction

Deriving fuel properties from spectroscopic signatures is not a new idea. Spectroscopy is a commonly used technique for determining the composition of materials as well as its physical and chemical properties. However, since fuels are typically composed of a mixture of multiple pure chemical components, each having unique properties, their presence and exact positioning in spectral regions of a fuel mixture are unknown. With years of research in this field, only a few of these components have been successfully connected to their placements in spectral regions. While chemical researchers continue to explore this problem from the domain standpoint, an alternative approach for determining fuel properties is being investigated as a resolution. This new direction combines the power of Machine Learning (ML) and chemometrics to build a robust data-driven approach for mapping spectral features and fuel properties to identify chemical components in mixtures. The spectroscopic data are typically limited to the number of physical mixtures produced, prepared, and processed by a researcher using a particular spectroscopy collection method. Therefore, collecting large numbers of observations is a time-consuming and expensive task, an issue that we address in this research proposing a synthetic data generation approach.

At present, Machine Learning (ML) is widely used in a variety of fields. It

refers to a group of computational and statistical techniques that maps a set of input characteristics to a response, with a priori unknown relationships, such that the estimated mapping (model) can be *utilized* for response prediction of unseen future observations.

There are a group of ML models with explicit functional form, either linear (e.g., Linear Regression (2)) or non-linear (e.g., *K*-Nearest Neighbor (3)). Several problems can be addressed by modeling linear relationships between a response value and a set of predictor variables that a human observing the process easily understands. In contrast, other problems require a more sophisticated mapping, leading to various non-linear ML modeling techniques. Within this group, a subset of more advanced ML modeling techniques has recently become more prevalent. Such techniques have no explicit functional form associated with the modeling, and instead, the mapping follows an architecture connecting input characteristics to output through a large set of nodes with a chain of transformations. Although such network-based models yield black-box complex functional form, they successfully capture the non-linearity and specific properties of the underlying unknown function. In reality, the nature of the underlying mapping is not known in advance.

Presently, machine learning has been successfully implemented to solve a range of problems from finance, engineering, and medicine to applied sciences. In some applications, these models achieve better performance than humans, such as object recognition, detection, and tracking (4; 5), beating world Chess and Go champions (6; 7), solving protein folding problems (8), etc. Unlike a human, however, a trained black-box model cannot provide reasons for a specific decision or choice, even though the results are often accurate, lacking a very important component of human behavior – explainability. Therefore, to claim that ML is truly capable of correlating the underlying relationship between spec-

troscopic data and material physical and chemical properties, it must be fully transparent for domain experts.

That being said, we employ the most advanced algorithms to explore the peaks and valleys of spectral data and accurately predict the ignition properties of fuel in real-time and at scale. Having access to a limited number of training samples and a large number of attributes results in a complex problem setting. Hence, we develop a framework that adopts a reduced subset of attributes for prediction instead of the entire set of attributes. This approach is two-folded, as it makes the prediction efficient and scalable in real-time and reduces the complexity to avoid overfitting issues on the unobserved noisy data. Furthermore, the reduced dimensionality of such a tool enables a more interpretable and accessible model for domain specialists to evaluate.

While the data collection is typically peripheral to constructing predictive models, it is essential to scale such models to solve real-world engineering problems. Thereby, we consider an end-to-end process from collecting the spectroscopic data to obtaining the results. The data collection often requires a robust, sizable sensor. Such sensors, however, can be miniaturized to capture only a selected region of spectroscopic signature if, and only if, there is enough information in those regions to differentiate between various types of fuels and their properties. Although the response of substrate fuel molecules is continuous over a specified light frequency range, the response is recorded in discrete steps due to instrument resolution limitations. Thereby, the underlying fuel chemical composition is represented by discrete feature locations on the wavelength axis. That being said, pinpointing these regions creates a pathway to optimize the data collection process, which is a requirement for the real-time deployment of such a sensor-to-prediction system as a whole.

There are several spectroscopy techniques used in real-time applications to



collect fuel spectroscopic data. The two datasets used in this work consist of fuel spectra collected using Raman and diamond Attenuated Total Reflectance (ATR) spectroscopy. These spectroscopic methods work on the light scattering principle, where the incident laser light is scattered by the sample's molecules to produce a unique spectrum. The spectroscopic data can be visualized as a series of data where each feature is recorded as the scattering intensity value occurring at a particular wavelength axis position. These techniques are frequently used in chemistry to provide a structural fingerprint by which substrate molecules can be identified. Explicitly, the fundamental Chemical Functional Groups (CFG) present in the sample are the main reason for producing unique chemical fingerprints of the sample under observation. As a unique structural fingerprint is acquired for each sample, fuel ignition quality such as Derived Cetane Number (DCN) can be measured for each sample and associated with its spectral data, thus providing a framework for predicting the DCN of unknown and unseen fuel samples.

DCN (i.e., CN) is one of the main indicators of fuel ignition quality. Similar to spectroscopic and CFG correlations, CN is highly correlated to the quantity and type of functional groups present in the fuel (9; 10; 11). Thus, feature selection is performed to extract wavenumbers important for CN prediction and, therefore likely correspond to CFG locations. This can be validated using the knowledge gained from the physical chemistry analysis, completing the human-in-the-loop approach.

Spectroscopy can suffer from several measurement accuracy issues: poor signal intensity, broad fluorescence baseline interference, and dark current noise. Although various pre-treatment and calibration techniques are often used before the analysis to ensure precise and consistent data readings, both spectroscopic data and measured CN values have an observable level of statistical noise. Bear-

ing that in mind, we particularly focus on models capable of extrapolating predictions for any and all unseen fuel samples.

Ultimately, we show that our end-to-end spectroscopy mapping and prediction framework is capable of identifying important features (wavenumbers) that are likely correlated with locations of chemical functional groups in the spectrum. Further, we show that using a reduced subset of important features to train the prediction model helps improve CN prediction accuracy, which is a measure of one of the underlying fuel properties. Therefore, the techniques used to map these relations remain transparent to users, and models are scalable for real-time deployment.

The biggest constraint, however, is the limited dataset size. Lack of diverse training dataset results in overfitting on the seen data and poor generalization ability to predict CN for unforeseen fuel samples. Moreover, spectroscopic data has an inherent noise associated with instrument resolution and collection process in a real setting where exogenous factors, such as room temperature, humidity, etc., affect laser power and result in noisy data. Therefore, to ensure that a robust prediction model is used for rapid data analysis in a real-time setting, it must be trained on a large and diverse dataset that considers various types of mixtures and above discussed noise levels. To that end, a simple perturbation of existing data will not be sufficient to produce new and well representative set of samples to train on.

As a resolution, we develop a generative technique to produce realistic, high-quality fuel spectra variations similar to ones that can be encountered in real-time. The main goal is to create a strong simulation core capable of generating quality synthetic spectroscopic data that complements the existing training dataset and improves overall data representation. As a secondary goal, by expanding our dataset in a meaningful way, we aim to enhance the prediction

performance.

To this end, we design a complex Generative Adversarial Network (GAN) model capable of learning the underlying distribution of given spectroscopic data. GAN is a popular, deep generative technique for improving data representation through the synthesis of new artificial samples that hold the same properties as original data. The use of GAN for boosting prediction performance spans many domains. The examples where generation of new data points helps to improve video frames prediction (12), risk prediction (13) or medical ailment classification (14) are just a few to name. A robust trained GAN model is then used to produce a variety of synthetic samples that are distinct from the original ones but follow the same distribution. Using additional domain information on real mixtures' known functional group concentrations, we further refine our model to target the production of synthetic spectra with predefined, desired properties. The evaluation of the quality of generated samples relies on estimating their similarity to the original training data that GAN was trained on, as well as expert evaluation to ensure synthetic spectra adherence to the chemistry domain. The former is achieved using statistical similarity tests, such as Kolmogorov–Smirnov (KS) statistic, which measures the similarity between real and synthetic cumulative distributions across all wavenumbers, and principal component decomposition analysis. The latter is achieved by producing targeted samples using conditional information to evaluate how well GAN learned a particular mixture mapping. Next, we employ a weak supervision approach to generate imperfect but plausible labels for our unlabeled synthetic dataset. This artificially generated and labeled dataset is then used for training a new CN prediction model in a supervised setting. The quality of the generated data is evaluated once more using the ML efficacy test, which determines whether substituting synthetic data with real data provides a comparable prediction er-

ror, indicating that the generated data are realistic and thus trustworthy.

In this research, we show that synthetic spectra generated with stable, trained GAN can address the limited spectroscopic data issue and boost dataset diversity. The quality of the synthetic data is assessed and found to be consistent with observable data. We present an improvement in prediction accuracy over baseline upon retraining prediction models on the new synthetically-expanded dataset, further contributing to our goal of building a powerful, robust and explainable end-to-end spectroscopy mapping and prediction framework. With the entire process grounded in domain-expert approval, we achieve clear empirical proof that the map between fuel properties and sample spectrum representation can be generated using ML technologies. From feature selection to synthetic data generation, we show that connections that exist between materials and their digital fingerprints can be learned and exploited. Given the limited size of fuel spectroscopy data, comprehensive learning and ignition property prediction model can be developed.

In Chapter §2 we begin with analyzing the current literature on machine learning, the role of spectroscopic data analysis in chemometrics, and the significance of machine learning in spectra data analysis. We further discuss feature selection and explainable AI techniques research and how these methods help to interpret complex models. Next, we review popular predictive modeling, feature selection and interpretation techniques. In Section § 2.2.1 we provide background on popular and successful machine learning models, particularly in the chemometrics domain, including Support Vector Machines and Neural Networks. In Section § 2.1.1 we review feature selection methods which help us identify important spectroscopy features and provide details on interpretation techniques used to explain the behavior of prediction models. In Section § 2.3 we focus our attention on the practical implementation of these methods

and include details about the deployment challenges in the real-time setting. In Section § 2.3.5 we provide experimental results and discuss our findings.

In Chapter §3 we shift our attention to addressing the issue of the limited size of spectroscopy data. We begin by discussing popular data augmentation and data synthesis approaches in §3.1 and cover existing literature on using generative modeling, particularly GAN, for spectroscopy data synthesis in §3.1.1. In §3.2 we review common GAN variants, their advantages and disadvantages, and underlying technical principles. We then provide details on evaluation methods used throughout the rest of the Chapter to establish the quality of the generated samples. Moving to §3.3, we discuss step by step implementation of GAN to generate realistic artificial spectroscopy that holds information about the physical properties of real fuels. After obtaining the desired quality of generated data, we perform a final domain-expert evaluation, generating an expanded spectroscopy dataset, and investigate the significance of using a new dataset for boosting prediction model accuracy. We then summarize the results of our research in §4.

# Feature Selection

The majority of this chapter's content are adopted from our recently published paper, (1).

## 2.1 | Background

To interpret the ML decision process, there exists three groups of *decomposition-based*, *model-based* and *model-agnostic* techniques, whose advantages and disadvantages are summarized in Table 2.1. The widely used decomposition techniques such as Principal Components Analysis (PCA) (15), and Partial Least Squares (PLS) (16) with its variants, such as Interval PLS (17), Forward and Backward Interval PLS (18), Moving Window PLS (19) and few others, are preprocessing methods that can be used to reduce the input space dimensionality before the training stage. The model-based interpretability is focused on constraining the structure of ML models so that they readily provide useful information about the uncovered relationships. "As a result of these constraints, the space of potential models is smaller, which may sacrifice training predictive accuracy" (20) to achieve a better generalization performance. Linear Regression (LR) (2), Logistic Regression (21) and Decision Trees (22) are innately interpretable.

Therefore, analyzing the interactions between features or between features and response within such models enables explaining the model outcome. On the other hand, the model-agnostic methods can be used on any ML model and are usually applied post hoc (post-model) following the model training stage.

Local and global interpretation methods are two types of model-agnostic interpretation methods (23). Global methods explain the on-average effect of features on the final prediction outcome. Local techniques, however, seek to explain particular predictions. The latter group of techniques is more useful when we do not have access to a lot of data and want to explain the behavior of the model for every single instance of the data. Most of the local model-agnostic interpretable techniques require a “surrogate or a simple proxy model that can be applied to learn a locally faithful approximation of a complex, black-box model based on outputs returned by the black-box model” (24). This approach is also known as knowledge distillation (25). Alternative to local interpretation that helps explain individual prediction, are global methods that describe entire model behavior across all predictions. Global methods offer “transparency about what is going on inside a model on an abstract level” (26). The advantage of Model-agnostic techniques is their flexibility to explain any model, providing consistency in explanation across various prediction methods. Therefore, we can select certain attributes that affect the target outcome using these interpretable methods, which will be summarized later. As mentioned above, some models are interpretable by nature, and some are complex black-box. The biggest advantage of model-agnostic techniques is their ability to explain such complex models. Moreover, this advantage can be extended from complex models to simpler ones as well.

Interpretability	Models	Advantage	Disadvantage
Decomposition-based	PCA	Computationally efficient. Removes correlated features.	Loss of direct mapping from features to output. Information loss due to improper number of components.
	PLS	Estimates correlation between features and target variable. Calculations are fast	More prone to overfitting on limited datasets. Assumes linear relationship between features and response
Model-based	RF	Ability to provide ranked feature importance. Handles missing values. Works well on high-dimensional data. Less biased towards more important attributes.	Validity dependant on sample size (sample bias). Computationally complex.
	Ridge	Ability to address multicollinearity issue. Shrinks unimportant variables.	Assumes linear relationship between predictors and target variable. Requires penalty parameter tuning. Low sparsification ability.
Model-agnostic	SHAP	Global interpretations are consistent with the local explanations. Solid theoretical foundation. Ability to provide local interpretations.	Computationally inefficient. Ignores feature dependence. Unreproducible interpretations. Live explanations require access to data.
	GS	Intuitive and straightforward. Ability to use any model as surrogate. Ability to measure surrogate models' performance to approximate black box predictions.	Draws conclusions about model and not data. Surrogate model interpretations can be not equally good for all subsets of data.
	LIME	Works for any data type. Explanations are selective and contrastive. Provides fidelity measure to estimate explanation reliability.	Instability of explanations, as repeated explanations differ. Identifying correct sampling neighborhood is imperative. Unreproducible interpretations.

Table 2.1: Summary of feature selection methods, their advantages and disadvantages.



While the body of research on spectroscopic analysis is extensive, to the best of our knowledge, there is no single comprehensive work focused on developing both explainable and scalable predictive models for spectroscopic data. Most publications in the field either solely focus on obtaining prediction, for example, applying popular ML methods for octane prediction using infrared spectroscopy (27), or the use of common feature elimination techniques (28) to improve the prediction accuracy. The implementation of explainable black-box models is limited to interpreting functional near-infrared spectra data in developmental cognitive neuroscience using simple multi-variate analysis (29) and using Local Interpretable Model-Agnostic Explanations (LIME) (30) on optical emission spectroscopy of plasma (31).

Therefore, we investigate the performance of a wide range of successful predictive models and implement model-based and model-agnostic interpretable techniques to achieve at-scale models for real-time practice. We consider three predictive models, from basic Linear Regression to non-linear Support Vector Machine (SVM) (32) and network-based Neural Network (NN) (33) regressions. Furthermore, we consider PCA, PLS, Ridge Regression (34) and Random Forest (35) for model-based feature selection methods, as well as popular local model-agnostic interpretable techniques, such as LIME and Shapley Additive Explanation (SHAP) (36), and Global Surrogate (GS) (37) as global model-agnostic method.

Performing a set of comprehensive experiments, we discuss the set of tools for spectroscopy data analysis beyond what has been covered in literature so far and provide informative insights on their challenges in a high-dimensional and limited spectra data setting. To provide a precise evaluation of feature selection techniques, we propose two metrics: Correctness, which measures selected features' adherence to known chemistry using domain expertise, and Perfor-

mance, which measures testing error on a predicted CN value. Additionally, we construct a trade-off scale between Correctness and Performance to evaluate the overall accuracy of the above-mentioned techniques in identifying such attributes.

In Section § 2.1.1, we analyze the current literature on machine learning, the role of spectroscopic data analysis in chemometrics, and the significance of machine learning in spectra data analysis. We further discuss feature selection and explainable AI techniques research and how these methods help to interpret complex models. Next, we review two major components of our work: predictive modeling and feature selection and interpretation techniques. In Section § 2.2.1 we provide background on popular and successful machine learning models, particularly in the chemometrics domain, including SVM and NN. In Section § 2.1.1 we review feature selection methods which help us identify important spectroscopy features and provide details on interpretation techniques used to explain the behavior of prediction models. In Section § 2.3 we focus our attention on the practical implementation of these methods and include details about the deployment challenges in the real-time setting. Finally, in Section § 2.3.5 we provide experimental results and discuss our findings.

### 2.1.1 | Related Works

Spectroscopic techniques have been widely used for different purposes in various domains such as petrochemical (38; 39), medical, pharmaceutical, and biological (40; 41; 42), food and agricultural (43; 44; 45; 46), engineering (47) and material and geologic (48; 49) analysis to monitor reactions and conditions of a final product.

There is an extensive literature in the petrochemical industry and the chemo-

metrics discipline that outlines various methods to predict fuel Cetane or Octane Number. The relationship is often established using information from fuel Quantitative Structure Property Relationships analysis (50), Fatty Acid Methyl Esters composition analysis (51) or, most commonly, spectroscopy analysis. There are too many forms of spectroscopy to mention, but some popular ones are: Gas Chromatography-Mass Spectroscopy (27; 52), Nuclear Magnetic Resonance Spectroscopy (9), Fourier-transform Infrared (FTIR) Spectroscopy (53; 54; 55; 56) and Raman Spectroscopy (56; 57; 58). Using infrared (IR) technique is arguably one of the most popular spectroscopy techniques, with some researchers focusing on probing Near Infrared (NIR) (28; 46; 57; 59; 60) or Mid-Infrared (MIR) regions. However, Raman Spectroscopy provides a big advantage over IR Spectroscopy since it allows collecting spectra in similar IR regions without major preparation or damage to the sample. Due to its shorter runtime, we choose Raman spectroscopy to further aid the goal of real-world deployment of our evaluation framework.

Summarizing the existing body of work on spectroscopy analysis (Table 2.2), it is worth mentioning that the majority of studies on fuel spectroscopy do not collect their own data or have a limited dataset size. The number of available fuel samples is likely a major limiting factor in producing a large, diverse dataset. While spectroscopy is arguably the fastest, cheapest, and least destructive way of gaining insight into materials, the overall data collection process for spectroscopy analysis, from fuel mixture creation, equipment setup, calibration to the final reading, is still a time-consuming task. While the majority of the literature covers fuel ignition qualities prediction, the choice of the predictive model is often limited to either linear or non-linear methods, and rarely are both compared. Additionally, fewer topics are dedicated to extracting or explaining (see Interpretability column in Table 2.2) spectroscopic features. To this end, most

researchers apply model-based feature selection techniques to identify and remove noisy features in order to improve prediction accuracy, and computational efficiency (27; 28; 56; 57; 59; 60; 61; 62; 63; 64; 65; 66; 67; 68) and avoid discussion on model scalability and interpretation on a noisy test set. Furthermore, the existing literature mainly focused on model-based and decomposition-based techniques that consider global importance, which is less effective for limited data settings. In another body of work, popular feature selection methods such as PCA (69), and PLS (70) were employed to discover the correlation between decomposed fuel spectra and fuel sample clustering results (53), help isolate certain chemical groups responsible for the deviation in predicted values (27; 46), and correlate certain spectra regions of pharmaceutical tablets to the concentration of antiviral drug (60).

Work	Collected Data	Linear Models	Non-linear Models	Model-based FS	Model-agnostic FS	Scalability	Interpretability
(50)	-	-	+	-	-	-	-
(61)	-	-	+	-	-	+	-
(54)	-	-	+	-	-	-	-
(57)	+	+	-	+	-	+	-
(39)	+	+	-	-	-	-	-
(71)	+	+	-	+	-	-	-
(60)	+	+	-	+	-	+	+
(58)	+	+	+	-	-	-	-
(72)	+	+	+	-	-	-	-
(73)	+	+	-	+	-	-	-
(53)	+	-	+	+	-	-	+
(62)	-	+	+	+	-	+	-
(52)	+	+	+	-	-	-	-
(46)	+	+	-	+	-	-	+
(27)	-	+	+	+	-	+	+
(63)	+	+	-	+	-	+	-
(55)	+	+	-	-	-	-	-
(74)	+	+	+	-	-	-	-
(38)	-	-	+	-	-	-	-
(51)	-	+	+	-	-	-	-
(56)	+	+	-	+	-	+	-
(28)	-	+	+	+	-	+	-
(59)	-	+	+	+	-	+	-
(75)	+	+	+	-	-	-	-
(29)	-	+	-	+	-	-	+
(76)	+	-	+	-	-	-	+
(31)	-	-	+	-	+	-	+
(64)	+	+	+	+	-	+	-
(77)	+	+	+	-	-	-	-
(9)	+	+	-	-	-	-	-
This Work	+	+	+	+	+	+	+

Table 2.2: Summary of related work in comparison to this work

As previously discussed, few works focus on explaining learning and predictive modeling using spectra data. In (29) authors applied linear multi-variate analysis to interpret development cognitive neuroscience spectroscopy data. Direct visualization of gradient-weighted class activation mapping of Convolutional Neural Network was developed in (76) to interpret detection of volatile organic compounds.

Recently, model-agnostic methods have attracted a lot of attention for feature evaluation, such as Shapley Additive Explanation (SHAP) (36) and LIME (30). Explainable AI techniques in general have been widely used to explain predictions in financial and chemical time-series data (78; 79; 80; 81) vibrational-based Structural Health Monitoring signals (47), hyperspectral imaging (82) and electrocardiogram data (83). However, to the best of our knowledge, only one recent work focused on using the model-agnostic method (LIME) to explain the non-linear predictions of spectroscopy data to characterize plasma solution conductivity (31).

After investigating the existing research, we develop an efficient, scalable framework that ensures prediction accuracy and transparency together. This chapter simultaneously investigates the performance of the most successful ML models in the literature, including SVM and NN, and covers a comprehensive examination of both model-based and model-agnostic explainable AI techniques for spectroscopy data analysis. The goal is to derive the most accurate, scalable prediction model specifically with limited data and in noisy settings. More importantly, we aim to ensure the interpretability and transparency of the prediction outcome to the human expert, supporting the fact that predictions are grounded in domain science and, therefore, can be fully trusted to make further decisions. We also focus on modeling raw, unscaled, noisy data to ensure it can be deployed fast in the real world without any major preprocessing. Finally, we

promote feature selection techniques that are able to determine local explanations rather than global to provide deeper insights on the impact of attributes on the response when the data are limited.

## 2.2 | Methodologies

### 2.2.1 | Predictive Modeling

Machine Learning is referred to a statistical tools encoded in a machine to make predictions about future observations based on historical data. There are many ways to categorize ML methods; supervised and unsupervised, linear and non-linear, etc. In this paper, we group them into two categories: interpretable and non-interpretable (black-box), with Linear Regression (2) being an example of the former one, and Support Vector Machines (32), and Neural Networks (33) being an example of the latter. As discussed in § 2.1.1, the two popular methods in chemometrics are PLS Regression (16), and PCA Regression (15), as they can first provide insight into important features in high-dimensional data by decomposing it into latent structures using PCA and PLS, and build interpretable linear regression models on top for prediction.

There is a long tradition of employing linear models in chemometrics. As more recent examples, Jameel et al. used multiple LR and nuclear magnetic resonance spectroscopy to predict fuel ignition quality (9). Barra et al. used PLS regression with FTIR spectroscopy to predict cetane number in diesel fuels (55). Balabin et al. also compared the performance of PLS and PCA regression models with NN while analyzing biodiesel properties using near-infrared (NIR) spectroscopy (64). However, the main drawback of any linear model is its inability to capture complex non-linear relationships within the data, especially

when the access to the training data are limited. Therefore, the use of Support Vector Regression (SVR) and deep learning models has been particularly prominent in chemometrics. Using SVR, Kiefer et al. achieved superior results on Raman spectroscopy of algal production of complex polysaccharides over LR (71), while Alves et al. noted that SVR outperformed PLSR for NIR spectra analysis (72). Balabin et al. has also explored the deployment of NN and SVR for analytical chemistry and concluded that not only SVR outperforms PCA and PLS regression methods (64), but that SVR also performs similarly to NN, with SVR having the advantage in producing a more generalized model capable of efficiently dealing with non-linear relationships (75). Similarly, NN is also capable of capturing unique spatial features and have been shown to perform well on spectroscopy data analysis (27; 38; 52; 53), given a wealthy amount of data. In our work, we investigate the performance of LR, SVR, and NN models to process high-dimensional spectra data for prediction and explainability effort.

### 2.2.1.1 | Support Vector Regression

Support Vector Machine is a supervised learning model grounded in Vapnik–Chervonenkis computation learning theory (84), which explains the learning process from a statistical point of view, ensuring high generalization ability on unseen data. SVM solves both classification and regression problems by identifying an optimal separating hyperplane with maximum margin to the training observations, formulated as a convex optimization problem. In a regression setting, the optimal hyperplane is the decision surface that best approximates the continuous-valued function (85). The goal is to first arrive at a minimized convex loss function that produces an error in predicted values at most equal to the specified margin, called the maximum error  $\epsilon$  (epsilon). At the same time, the decision surface must stay as flat as possible while containing most of the training



samples (86). An important property of SVR is its ability to map input vectors to a high-dimensional feature space where a non-linear decision surface can be constructed that fits the data within a threshold of values within a specified margin. Since the data can often not be separable in initial finite-dimensional space, mapping it into a much higher-dimensional space, aka kernel space, makes the separation easier.

The optimization, which has a unique solution, is further solved, and since not all points are going to fall within the margins, slack variables  $\xi_n$  and  $\xi_n^*$ , which denote deviation from the margin, are introduced to deal with otherwise infeasible constraints. The constant  $C$  is then introduced to impose a penalty on observations that lie outside the margin to prevent overfitting and determine the trade-off between the flatness and amount of deviation that can be tolerated. Ultimately, the decision surface is confined using support vectors, which are the most influential instances that lie outside the tube boundaries and affect its shape. In order to minimize the computational complexity of the described optimization problem, it is solved using Lagrange dual formulation. Mathematically speaking, given the set of observations  $\mathbf{X}$  where each  $X_i \in \mathbb{R}^M$ , for  $i = 1, \dots, N$  and where  $N$  is the number of samples and  $M$  is the number of features (dimensions), with  $y_i$  being the predicted value, we can express the optimization problem as:

$$\min \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^N \xi_i + \xi_n^* \quad (2.1)$$

s.t.

$$y_i - \mathbf{w}^T \varphi(X_i) \leq \varepsilon + \xi_i^* \quad i = 1, \dots, N \quad (2.2)$$

$$\mathbf{w}^T \varphi(X_i) - y_i \leq \varepsilon + \xi_i, \quad i = 1, \dots, N \quad (2.3)$$

$$\xi_n, \xi_n^* \leq 0, \quad i = 1, \dots, N \quad (2.4)$$

Where  $\mathbf{w}$  is the weight vector of the separating hyperplane and  $\varphi(\cdot)$  is a transformation function, i.e. kernel, that maps vector  $\mathbf{X}$  to a high-dimensional space, that computes inner products of the input vectors. Using kernel, or kernel trick, pairwise similarity comparisons between training data observations are used instead transforming data to avoid extremely high number of combinations. Two popular kernel functions include: Linear dot product  $\varphi(X_i, X_j) = \langle X_i X_j \rangle$  and Polynomial  $\varphi(X_i, X_j) = (\gamma \langle X_i, X_j \rangle + Coef0)^d$ , for  $d > 0$  where  $\gamma$  is a scale factor that defines how a support vector shapes the decision surface, and  $Coef0$  is an independent term used to overcome dot product computation issues for high-dimensional data. The parameters are further covered in Section § 2.3.3.

### 2.2.1.2 | Neural Networks

NN is a computing system that is commonly used for supervised learning. It is represented by a network of artificial neurons or nodes connected by links, where each link has an associated randomly initialized weight and activation level. Each node has an input function (typically summing over-weighted inputs), an activation function, and an output. The weights are updated through a forward and backward propagation until they converge to optimal estimates.

Through the forward approach, the input function of each unit is passed through the activation function, typically a non-linear function, and transformed to a new value that would be passed to the nodes in the subsequent layer. This process is known as *forward propagation*, during which a network learns and creates its own features. Mathematically, the operations in input layer are shown in Equation 2.5, where  $a^1$  is input layer activation, expressed as function  $g$  of weights  $\mathbf{w}$  and training data  $\mathbf{X}$ . Working forward through the network, the input function of each unit is applied to the weighted sum of the activation on the links feeding into that node. Forward propagation ends at the final (output) layer  $L$ , which produces a value based on function  $h_w(X)$  of all previous layer transformations (Equation 2.6). Lastly, the total prediction error is calculated using problem-specific loss (or cost) function, e.g. mean squared error (Equation 2.22), cross-entropy (Equation 2.7), so that the gradients which are used to update the weights in next step can be derived.

$$\mathbf{a}^1 = g(\mathbf{w}^1 \mathbf{X}) \quad (2.5)$$

$$h_w(X) = \mathbf{a}^L = g(\mathbf{w}^L \mathbf{a}^{L-1}) \quad (2.6)$$

$$J(\mathbf{w}) = (1 - y)(\log(1 - \hat{y}) + y \log(\hat{y})) \quad (2.7)$$

Each node  $j$  in layer  $l$  is “responsible” for fraction of the error  $\delta_j^l$  in output nodes it is connected to. Hence, through the backward approach, the error associated with each unit from the preceding layers are back-calculated following Equations 2.8- 2.9 and the contributing weights are adjusted, accordingly. This process is known as *backpropagation*, during which partial derivatives of error

measurement are calculated to track gradient descent,  $\Delta^l$ , Equation 2.10 that minimizes cost function until convergence is reached.

$$\delta^L = \mathbf{a}^L - \mathbf{y} \quad (2.8)$$

$$\delta^l = (\mathbf{w}^l)^T \delta^{l+1} (\mathbf{a}^l * (1 - \mathbf{a}^l)) \quad (2.9)$$

$$\Delta^l = \Delta^l + \delta^{l+1} \mathbf{a}^{(l)T} \quad (2.10)$$

The choice of network hyperparameters determines the network architecture and how the network is trained and, therefore, is crucially important to obtain a high-performing model. The most common methods of hyperparameter selection are Grid Search, Random Search, and Bayesian Optimization. In this paper, we will adopt Bayesian Optimization to identify the optimal NN architecture. There has been several activation functions proposed in the literature including the most common ones such as Sigmoid and Relu.

## 2.2.2 | Decomposition-based Feature Selection Techniques

### 2.2.2.1 | Principal Component Analysis.

PCA is an unsupervised dimension reduction method that transforms data to a new coordinate system with a reduced set of variables that retains most of the information from the original space. The transformed space can be imagined as  $P$ -dimensional ellipsoid, where longest axis of an ellipsoid represent direction of maximum variance within the data for  $P < M$ , where  $M$  is the number of original features (dimensions). Given  $N$  number of samples and an  $N * M$  matrix of data  $\mathbf{X}$ , where  $X_i \in \mathbb{R}^M$ , for  $i = 1, \dots, N$  and  $j = 1, \dots, M$ , the data has to

be standardized to be centered around 0 by subtracting the mean value of each variable from individual data points in a given dimension. The key idea behind PCA is to combine original variables into a set of latent vectors  $\mathbf{Z}$  in a linear way, where  $Z_p \in \mathbb{R}^M$  for  $p = 1, \dots, P$  same as  $\mathbf{X}$  since it is its linear combination. More formally, the latent vectors can be presented as follows:

$$Z_p = \sum_{j=1}^M w_j X_j \quad (2.11)$$

The latent variables are constructed sequentially. The first projection  $\mathbf{z}_1$  can be written as:

$$Z_1 = w_1 X_1 + \dots + w_j X_j, \quad (2.12)$$

Where  $\mathbf{w}$  is the vector of weights constrained so that its sum of squares is equal to 1. According to (87), since the matrix  $\mathbf{X}$  contains variation relevant to the problem, it seems reasonable to have as much as possible of that variation also in  $\mathbf{Z}$ . Suppose this amount of variation in  $\mathbf{Z}$  is appreciable. In that case, it can serve as a good summary of the  $\mathbf{X}$  variables, hence, allowing us to reduce the number of variables used in the original space. The problem is therefore constructed as maximizing the variance of  $\mathbf{Z}$  with respect to the optimal weights. By substituting (Equation 2.12) into mathematical notation for returning the argument  $\mathbf{w}$  of the maximization function and rewriting it in a vectorized format, we obtain the following objective function:

$$w_p = \arg \max_{\|\mathbf{w}\|=1} (\text{var}(\mathbf{Z})) = \arg \max_{\|\mathbf{w}\|=1} (\mathbf{w}^T \mathbf{X}^T \mathbf{X} \mathbf{w}) \quad (2.13)$$

Using the result of the covariance matrix,  $\mathbf{X}^T \mathbf{X}$ , we find that the optimal  $\mathbf{w}$  are therefore the eigenvectors of  $\mathbf{X}^T \mathbf{X}$ , where the first ordered eigenvector corresponds to  $w_1$  and so on. Scaled by the variance, which is squared root of

corresponding orthogonal eigenvalues, columns of  $\mathbf{w}$  are therefore called loadings in PCA. As a result, reconstructed back to the original space and applied to fuel spectroscopy, the output of the PCA model contains information on scores of various features that contribute most to the explained variance in recorded absorbance or scattering intensity values.

### 2.2.2.2 | Partial Least Squares

PLS is a supervised alternative to PCA and is arguably the most widely used technique in the chemometrics domain. Similar to PCA, it identifies a set of features that are linear combinations of the original features. Unlike PCA, however, PLS considers the response  $\mathbf{y}$  and finds the multidimensional coordination of the feature space  $\mathbf{X}$  that contributes to most variance in the response  $\mathbf{y}$ . Essentially, partial least squares seek directions that have high variance and have a high correlation with the response, in contrast to principal components regression which keys only on high variance (88). After standardizing the data to have 0 mean and variance of 1, both  $\mathbf{X}$  and  $\mathbf{y}$  are decomposed as a product of a common set of orthogonal vectors and a set of specific loadings (89). The data and response matrices are decomposed as:

$$\begin{aligned}\mathbf{X} &= \mathbf{Z}\mathbf{P}^T \\ \mathbf{y} &= \mathbf{Z}\mathbf{Q}^T\end{aligned}\tag{2.14}$$

Where  $\mathbf{Z}$  is a matrix of latent vectors, and  $\mathbf{P}, \mathbf{Q}$  are the loading coefficient matrices. A rank regression is then performed to construct a matrix of latent components  $\mathbf{Z}$  as linear transformation of  $\mathbf{X}$ , where  $\mathbf{w}$  is a vector of weights:

$$\mathbf{Z} = \mathbf{X}\mathbf{w}\tag{2.15}$$

The idea behind PLS is to perform decomposition so that the information from both  $\mathbf{X}$  and  $\mathbf{y}$  is taken into account.

The elements of the weight vector  $\mathbf{w}$  are defined such that the squared sample covariance between response and the latent components is maximal under the condition that the latent components are mutually uncorrelated (90). Finally we adopt the following objective function to find optimal set of weights for each latent vector  $p = 1, \dots, P$ , that we later use for feature selection:

$$w_p = \arg \max_{\mathbf{w}} (\mathbf{w}^T \mathbf{X}^T \mathbf{y} \mathbf{y}^T \mathbf{X} \mathbf{w}) \quad (2.16)$$

## 2.2.3 | Model-based Feature Selection Techniques

### 2.2.3.1 | Random Forest

Random Forest is an ensemble learning method, which constructs multiple decorrelated decision trees trained on different subsets of the data and subsets of selected attributes to reduce the variance in  $\mathbf{y}$ . Although Random Forest is a prediction tool, it is widely used to rank the importance of the variables based on the number of times they are used during node splitting. For each node  $t = 1, \dots, K$ , where  $K$  is total number of nodes within a binary tree  $r = 1, \dots, T$ , where  $T$  is number of trees of the random forest, the optimal split in a classification setting is determined by impurity, measuring how well a potential split separates observations that are similar to each other (91). In regression problems, the measure of impurity ( $i(t)$ ) is variance of the predicted value of observations within each partition. Therefore, the importance ( $w$ ) of a feature ( $j$ ) is computed as the (normalized) total reduction of variance ( $\Delta i$ ) brought by that

feature across all trees. Mathematically it can be expressed as:

$$w_j = \sum_{r=1}^T \sum_{t=1}^K \Delta i_j(t, r) \quad (2.17)$$

### 2.2.3.2 | Ridge Regression

The linear regression regularization method is used to reduce model complexity by adding penalties to coefficients of variables in the cost function, such that the Residual Sum of Squares (RSS) loss function for  $i = 1, \dots, N$  and  $j = 1, \dots, M$ , where  $N$  is number of samples and  $M$  is number of data dimensions, takes the form:

$$RSS = \sum_{i=1}^N (y_i - \hat{y}_i)^2 + \alpha \sum_{j=1}^M w_j^2 \quad (2.18)$$

The first term in the above equation is the sum of squared error and the second term is the regularization component.  $\alpha$  is the penalty parameter used over all weights of features  $\mathbf{w}$  to shrink the magnitude of the unimportant ones to ensure that the model does not overfit.

Tuning  $\alpha$  hyperparameter controls the strength of the penalty term or essentially the amount of feature shrinkage, which results in sparse models with less number of parameters, easier to analyze than high-dimensional data models.

## 2.2.4 | Model-agnostic Feature Selection Techniques

### 2.2.4.1 | SHAP

SHAP stands for Shapley additives Explanations, which is considered a popular state of the art in explaining black-box machine learning models. SHAP is a technique to calculate the impact of each feature on the prediction outcome using Shapely Values. Shapley values were introduced in the 1950s by Lloyd



Shapley (92), who introduced it as a solution concept in cooperative game theory. The main idea is that any model output does not rely only on one single feature but on the entire set of features in the data set.

Suppose that we have a predictive model, where the game outcome represents the model prediction, and the players in the game represent the features. Considering all possible coalition among the players (features) and their effect on the game (model outcome), each player contributes to the team's result. The sum of the contributions for each player from each possible coalition returns the value of the target variable (model outcome) given a particular feature. As a result, Shapely Values calculates the contribution of each feature to the target value, which is referred to as local marginal contribution or local Shapley Values. Repeating the same process using combinatorial calculus and retraining the model over all possible combinations of features, we can calculate all local Shapley Values for a specific feature. The average absolute value of the local Shapley Values can be used as a measure of feature importance.

More formally, let  $S$  be a subset of features that does not include the feature for which we calculate the importance. Let  $M$  be the full set of features. Given a model  $g(x)$  trained to predict  $f(x)$ , the marginal contribution of feature  $i$  to the model's prediction and accordingly to the  $f(X)$  is:

$$w_j = \sum_{S \subseteq M \setminus j} \frac{|S|!(|M| - |S| - 1)!}{|M|!} [g(S \cup j) - g(S)], \quad (2.19)$$

Where  $S \cup j$  is the subset that includes features in  $S$  plus feature  $j$  and  $S \subseteq M \setminus i$  indicates all sets  $S$  that are subsets of the full set of features  $M$ , excluding feature  $j$ .

As discussed by (23), “the computation time increases exponentially with the number of features. One solution to keep the computation time manageable is to compute contributions for only a few samples of the possible coalitions”. Lately,

Lundberg and Lee developed an algorithm for interpreting model predictions (36), which uses the Shapely Values to reverse-engineer the output of any predictive algorithm and identify features' contributions. SHAP approximates the conditional expectations of SHAP values by using a selection of background samples to reduce the computation time. By aggregation over multiple background samples, SHAP estimates values such that they sum up to the difference between the expected model output on the passed background samples and the current model output ( $f(x) - E[f(x)]$ ). Features that contribute the most to the difference between the expected model output on the passed background samples and the naive case prediction are chosen as important features by SHAP. SHAP method can be used to analyze the prediction for both classification and regression models.

#### 2.2.4.2 | LIME

The LIME explanation method was originally proposed by Ribeiro et al. in 2016 (30). The key idea of LIME is to locally approximate a black-box model by a simpler glass-box model such as Linear Regression or Random Forest, which is easier to interpret. Such an interpretable model must be locally faithful, meaning it must correspond to how the black-box model behaves in the vicinity of the instance being predicted. LIME works by perturbing any individual data point and generating synthetic data, which gets evaluated by the black-box model and is ultimately used as a training set for the simple model. The variables are perturbed by sampling from a normal distribution and doing the inverse operation of mean-centering and scaling the values according to the means and standard deviations in the original training set. The LIME is capable of explaining any model, and thus it is model-agnostic. The aim of LIME is to minimize a loss function  $L(f, g, \pi_x)$  that can be expressed mathematically in the following way:

$$\zeta(x) = \arg \min_g L(f, g, \pi_x) + \Omega(g) \quad (2.20)$$

Where  $f$  is the original model,  $g$  is an interpretable model, and  $\pi_x$  is the similarity kernel that measures the proximity of a new perturbed sample point  $z$  to the original data point  $x$ . Additionally, the  $\Omega(f)$  is referred to as a measure of complexity, opposed to interpretability. The Equation 2.20 demonstrates that perturbed samples are generated around  $x$  and weighted by  $\pi_x$  to approximate  $L(f, g, \pi_x)$ . Using this approximation LIME can explain local behaviour of the original model  $f$  and measure the relative error between the explanation  $\zeta(x)$  and the original model predictions.

### 2.2.4.3 | Global Surrogate

Similar to LIME, the global surrogate model is used to approximate the predictions of highly non-linear ML models with simpler, interpretable models. Since it is a global method, the surrogate model tries to mimic the function of the entire black-box model to understand its overall behavior. A global surrogate model does not require any information on how the original black-box model works and thus is considered model-agnostic. The process of training a surrogate involves obtaining the predictions of the black-box model on the training dataset  $X$ . Then, a selected interpretable model is trained on  $(X, \hat{Y})$  using black-box predictions as targets. The surrogate model can be any interpretable model, such as Linear Regression, Decision Tree, K-nearest neighbor, or any model that the coefficients could provide insights into the model behavior. The ability of the surrogate to capture the behavior of the black-box model is estimated by computing the error between surrogate and black-box predictions, typically using the r-squared score. A caveat of global surrogate models is that the performance

of the underlying black-box model in predicting the actual outcome plays no part in training the interpretable black-box model (23).

## 2.2.5 | Evaluation Metrics

### 2.2.5.1 | Correctness

In this paper, the goal of feature selection is to select a subset of features in the spectra data that are considered most informative or relevant to the predicted CN values while adhering to known chemistry. The selected subset of features can be further compared to the theoretical locations of chemical functional groups within the spectroscopy data known from the domain expertise. Incorporating human knowledge in the learning loop allows us to evaluate the faithfulness of the applied techniques and ensure that a model can be trusted. To evaluate the performance of the feature selection techniques discussed in Section § 2.1.1, we define a measure to calculate the alignment of the selected subset of features using different techniques with the selected features by the expert. We refer to this measure as *correctness*.

The selected wavenumbers are binned by unique centered intervals to represent their true location, which can be otherwise shifted by several wavenumbers due to instrument noise. The method Correctness is then calculated using simple Jaccard Similarity ( $J$ ) (93) between a given subset of binned features indicated by  $S_F$ , which is obtained from considered feature selection techniques, and expert-selected locations indicated by  $S_{Ex}$ . The method Correctness can also be measured by converting proportion  $J$  to percentage to obtain values between 0 and 100%, representing a percent match to all expert-selected locations. The Jaccard Similarity is calculated as follows:

$$J(S_F, S_{Ex}) = \frac{|S_F \cap S_{Ex}|}{|S_F \cup S_{Ex}|} = \frac{|S_F \cap S_{Ex}|}{|S_F| + |S_{Ex}| - |S_F \cap S_{Ex}|} \quad (2.21)$$

Although the performance of a predictive model after reducing the original feature space is important for at-scale implementation in real-time, the correctness of the feature selection techniques is inevitable for transparency. In §2.3.5 we provide numerical results for the correctness of different considered techniques.

### 2.2.5.2 | Performance

To aid the scalability and efficiency of the model deployment in practice, the model fitted on the reduced subset of features must be comparable in terms of performance to the model trained on full-spectrum and to a reduced model trained on an expert-selected subset of features. However, note that the reduced model is more efficient for the data collection process. Since fuel CN is measured and predicted as a real value, each model's performance is assessed using the Mean Squared Error (MSE) metric, which is commonly used in regression problems. MSE calculates the difference between the predicted and true response values. As a result, the model with a lower MSE value on the testing set is deemed to be more favorable, as it predicts values that are close to the ground truth. The MSE is formally represented as follows:

$$MSE = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2 \quad (2.22)$$

## 2.3 | Experimental Results

### 2.3.1 | Domain

In the Physical Chemistry and Combustion domain, it is well established that chemical functional groups are the building blocks of any hydrocarbon structure such as fuels. (94) defines a functional group as “any portion of a molecule composed of a group of atoms, which governs both its physical and chemical properties as well as chemical reactivity”. Practical fuels, such as gasoline, comprise a large number of hydrocarbon molecules, which in turn might contain different functional groups. Knowing the quantity and type of functional groups present in a compound allows researchers to determine its properties. One of such properties is fuel ignition quality, which indicates how easily the fuel will ignite. Fuel ignition quality is one of the most important properties that scientists from the applied combustion field have been working for over the years. It is critical to measure ignition quality and correlate it to the fuel’s functional groups. Each functional group gives rise to unique peaks in the spectra in their characteristic regions. Therefore, this paper aims to detect their location on the spectral band using explainable AI tools and feature selection techniques.

In the context of predicting fuel ignition properties, the task of feature selection is to select a subset of wavelengths that improves model interpretability (further called *correctness*) and at-scale prediction performance. In a model with higher correctness, selected features must represent locations of CFG that determine fuel ignition properties, making the overall process transparent to the human operator. Involving a group of domain experts, we choose 120 wavenumbers (95) that correspond to chemical bonds in different functional groups within our data and compare them to the outputs of the above-listed feature selection

techniques (Section § 2.1.1). Note that each functional group stretches over certain regions of spectra, which can be identified by multiple peaks in that region. The exact locations of these peaks are unknown, however, using the literature on pure components (95) that make up the mixtures, we can try to estimate their locations. Each peak is considered as a feature selected by the expert  $S_{Ex}$ .

Taking into account the uncertainty associated with expert features and the shift in wavenumber read that may occur during the spectroscopy collection process due to instrument noise, we generate “bins” that capture the discrete output of attribute selection methods and match it to the expected locations. In other words, we divide the entire spectroscopy range into equal intervals (bins), assign discrete wavenumbers to the intervals they fall into, and calculate the fraction of unique intervals that were selected by both expert and feature selection methods. Each interval is arbitrarily chosen to be five wavenumbers wide in order to account for two wavenumber deviations from its potential location, which can be substantially bigger in practice. The precision of such a procedure would be exceedingly poor and unreasonable from a practical standpoint if each wavenumber from two subsets was attempted to be mapped exactly one to one.

Furthermore, there is a need for miniaturized spectroscopic instruments for high-profile applications in collecting the respective spectra for detailed analysis. Feature selection opens up a pathway towards such miniaturization. By selecting the important subset of wavelengths over the entire range of the spectrum, which affects the prediction performance of the output variable, it is possible to group the features into regions of importance. Once these regions are known, the spectroscopic instrument can be miniaturized by selecting filters such that the data are only collected for the important subset of wavelengths. As the scope of the spectral range has reduced, a smaller instrument could be designed for the same application without trading off the prediction accuracy.

Also, another approach could be to use lower resolution instruments which would require smaller components and thus a smaller instrument.

### 2.3.2 | Dataset

The spectroscopy data used in this chapter was collected at UIC High-Pressure Shock Tube Laboratory using a Raman spectrometer. The entire dataset includes 245 observations from 49 unique fuel samples collected at various times. The first 145 observations (based on 29 unique mixtures replicated five times each) were collected at a single session over four hours nonstop, where the laser power stayed consistent with minimizing the instrument noise. The second 100 observations (based on another 20 distinct mixtures reproduced five times each) were collected over a two-hour period using the same settings in a separate session. Environmental circumstances (e.g., ambient temperature) are exogenous factors that can affect laser power and, thus, spectroscopy repeatability, that cannot be completely controlled in real-world setting. Thus, this separate data collection allows us to consider statistical noise in the data for our scenario analysis. During the collection of the old dataset's observations, the laser power was measured as 350.5 mW, and for the new dataset, it was measured as 364.2 mW.

For both collected datasets, the resolution was set to  $7.1 \text{ cm}^{-1}$ , with wavelength range between  $52.52$  and  $3712.89 \text{ cm}^{-1}$ . The Raman laser power was set at the maximum laser power setting to ensure consistent readings for all observations. Each sampled spectra data initially had 2048 features (intensity values at different wavelength locations) and a measured CN value as the response. The datasets were pre-processed to filter out highly noisy regions outside of wavelengths range  $[181.45, 3200.82] \text{ cm}^{-1}$ , resulting in 1562 features. Note that the fingerprint region normally starts at  $500 \text{ cm}^{-1}$ , however, in this paper, we aim



to investigate the presence of functional groups in the  $[181.45-500] \text{ cm}^{-1}$  region. We specifically trimmed  $[50.52-181.45] \text{ cm}^{-1}$  since no scattering intensity value is recorded in this region. Similarly, we removed the data in  $[3200.82, 3712.89] \text{ cm}^{-1}$  region since no spectral features indicating functional group activity were present.

For simplicity we call the first 145 collected observations as *old* dataset and the 100 observations as *new* dataset.

Three scenarios are constructed accordingly to observe the effect of statistical and data collection noise on feature selection and prediction accuracy that is inevitable in the real-time practice of ML for ignition delay. The *Control* scenario includes 145 old observations, and the *Mixed* scenario includes 245 old, and new observations merged, both split into training, testing and validation set 80/10/10. As for the *Real-time* scenario, 145 old data points are split into training and validation sets 80/20, while the new 100 points are used as the testing set.

Using the preprocessed dataset for each scenario the reduced subset of attributes is identified implementing considered feature selection techniques including **PCA**, **PLS**, **RF**, **Ridge**, **SHAP**, **GS**, and **LIME**. Each predictive model, that shall be elaborated in Section § 2.3.3, is then fitted on the reduced subsets for each scenario as well as full set of 1562 features of data. The former is referred to as *Full model* and the latter as *Reduced models*.

### 2.3.3 | Predictive Modeling For Spectra Data

To implement ML models for CN prediction there are hyperparameters associated with each considered models discussed in Section § 2.2.1 including **SVR** and **NN**. Due to the small overall dataset size and to avoid data splitting and

model training bias, the data are randomly split 30 different ways using a random seed generator during each model development. The performance of the models is assessed based on the mean ( $\mu$ ) and standard deviation ( $\sigma$ ) across 30 random executions. To choose optimal hyperparameters, we perform Bayesian Optimization (BO) using a predefined range of parameters for SVR (Table 2.3) and NN (Table 2.4) for both Full and Reduced models trained on various subsets of features for each scenario. We use epsilon-SVR from the libsvm (96) package for SVR modeling and Keras (97) sequential model package to construct our NN.

Parameters	Options
Kernel	['poly', 'rbf', 'sigmoid', 'linear']
Degree ( $d$ )	[1 : 4]
Gamma ( $\gamma$ )	[0.0001 : 1]
Coef0	[0.01 : 10]
C	[0.1 : 1000]
Epsilon ( $\epsilon$ )	[0.01 : 10]

Table 2.3: SVR hyperparameters considered for optimization

Parameters	Options
Activation (hidden)	['relu', 'sigmoid']
Number of hidden layers	[1 : 10]
Hidden units	min_value=32, max_value=8000, step=32
Activation (output)	['linear', 'sigmoid']
Optimizer	['adam', 'sgd', 'rmsprop']
Learning rate	[1e-4 : 1.0]
Kernel regularization	[0.0001 : 0.01]
Kernel weight initializers	['random_normal', 'glorot_uniform', 'he_normal']
Batch size	[32 : 100]
Epochs	[100 : 1000]
Architecture	['up', 'down', 'up-down', 'down-up', 'random']

Table 2.4: Keras Neural Network hyperparameters considered for optimization

For SVR, the regularization hyperparameter  $C$  is a free parameter that trades

off the influence of higher-order versus lower-order terms in the polynomial.  $\text{Gamma}(\gamma)$  is a scaling parameter that controls the shape of Support Vector curvature, allowing it to fit the peaks observed in our data.  $\text{Epsilon}(\epsilon)$  is a margin term that allows more points to be included in the decision surface without penalizing them during the training. The choice of the kernel in SVM determines the shape of the transformed high-dimensional hyperplane and allows to avoid complex calculations, while the parameters  $\text{Degree}(d)$  and  $\text{Coef0}$  are typically used for the polynomial kernel to determine the degree of polynomial fit and adjust the independent term accordingly. It is worth mentioning that the (implicit) feature space of a polynomial kernel is equivalent to that of polynomial regression, but without the combinatorial blowup in the number of parameters to be learned (98). The algorithm used in epsilon-SVR calculates the outer product of two vectors of features (or a vector with itself), which can be used as an approximation of the polynomial kernel feature space instead of explicitly computing the outer product, which can be extremely inefficient. The resulting kernel space has the same dimensions as original training data, while full third-degree polynomial expansion of 1562 features would result in over 620 million features.

The optimal hyperparameters of SVR for both Full and Reduced models are obtained using open source Bayesian Optimization tool (99) as:  $\text{Kernel}='poly'$ ,  $\text{Degree}(d)=3$ ,  $\text{Gamma}(\gamma)=0.7$ ,  $C=0.7$ ,  $\text{Coef0}=0.1$ ,  $\text{Epsilon}(\epsilon)=0.1$ , with maximum number of iterations of the solver fixed to 100,000. This model is therefore referred to as BO-Tuned SVR.

The complexity of the SVR model is defined based on the number of support vectors defining the decision boundary. Table 2.5 illustrates the complexity of the SVR model for each scenario. Since the reduced subsets are created based on a smaller subset of individual wavenumbers to capture the complexity of the

spectra data, the model needs a larger number of support vectors to define the decision boundary. For example, in Table 2.5 we observe that 50 support vectors are selected for the Full model setting, and on average, 63 support vectors are necessary for the Reduced setting in a Real-time scenario. Moreover, the number of support vectors increases from 50 in the Control scenario (145 observations) to 113 vectors in the Mixed scenario (245 observations) for the Full model setting, where the noisy observations were included. This is also reflected in the average runtime shown in Table 2.7 as training time increases drastically for SVR with the addition of new noisy samples<sup>1</sup>. For example, the average training time for the BO-Tuned SVR model in Control scenario is 0.17 seconds, compared to 1.4 seconds in the Mixed scenario.

Since our goal is to ultimately deploy a strong predictive model in real-time, it requires a certain level of generalization to ensure accurate prediction of new, previously unseen observations. The SVR hyperparameter  $Epsilon(\epsilon)$  plays a significant role in the generalization power of the model. While BO maximizes exploitation of training data distribution, fine-tuning this parameter allows the construction of decision surfaces both accurate in shape and wide enough to generalize to unseen fuel spectroscopy samples, as will be shown in the Real-time scenario when test data are noisy. Using softer epsilon margin ( $Epsilon(\epsilon) = 0.66$ ) results in a reduced number of support vectors (33 compared to 50 for Full model in Real-time scenario), which further simplifies the model and makes computation more efficient (Figure 2.3a), aiding our scalability effort. More importantly, fine-tuned SVR drastically improves the performance of SVR for our Real-time scenario, in some cases decreasing testing error by the factor of 10.

When constructing Neural Network, the following hyperparameters are con-

---

<sup>1</sup>Note that the values represented under column "Time (s)" is the total computation time of 30 executions.

	# of Support Vectors			
	Control	Mixed	Real-time	
Full Model	50	113	50	BO-Tuned SVR
Expert	52	107	52	
PCA	73	126	73	
PLS	79	132	79	
RF	64	122	64	
Ridge	66	122	66	
SHAP	61	121	61	
GS	63	128	63	
LIME	48	119	48	
Full Model	33	67	33	Fine-Tuned SVR
Expert	28	63	28	
PCA	36	77	36	
PLS	53	88	53	
RF	34	75	34	
Ridge	28	60	28	
SHAP	37	68	37	
GS	33	66	33	
LIME	32	70	32	

Table 2.5: SVR Complexity

sidered. *Kernel weight initializer* determines the distribution of weights associated with each layer before the commencement of training and their consequent update through backpropagation. The *number of hidden layers* and *Hidden units* determines the power of the network to perform a linear or non-linear transformation on inputs and guides over model complexity. An *Activation* function in a neural network defines how the weighted sum of the input is transformed into an output from a node or nodes in a layer of the network (100). The *Kernel regularization*, also known as weight decay, is aimed at reducing the likelihood of model overfitting by keeping the values of the weights and biases small. *Optimizer* is an algorithm used to adjust model parameters (weights) to maximize

a selected loss function (in our case, it's *mean squared error*), while the *Learning rate* determines the rate of adjustment. *Batch size* defines a number of samples to propagate through the network at one time, and *Epochs* is a measure of the number of cycles it takes to train the network with all training data.

We select optimal hyperparameters for Full and Reduced models using Keras-Tuner (101) framework with the objective function aimed at reducing validation Mean Squared Error (MSE). The MSE mathematical formula is provided in Equation 2.22. The global random seed is set to ensure consistent kernel weight initialization and reproducible results. We also include an architecture-specific hyperparameter that dictates whether the overall architecture shape expands/shrinks or is produced at random in the optimal setting, for each consecutive layer of the network. We introduce a set of constraints to limit the exponentially large solution space of hyperparameters and make the optimization computationally stable while exploring versatile architectures. The number of nodes and layers are given the flexibility to be chosen at random or follow the pattern where the number of nodes is doubled or reduced by a factor of 2 for each subsequent hidden layer and constrained to a maximum of 8,000 nodes. The widening and shrinkage of the network can be both symmetric or asymmetric, with respect to the number of layers before and after the layer with minimum/maximum number of nodes. In an asymmetric case, the number of layers and nodes is selected randomly after picking layers with minimum/maximum nodes.

Given the budget of 1000 trials for BO, the optimal Full model architecture is determined to consist of the input layer, one hidden layer, and a final output layer with 1562-5984-1 nodes. The optimal Reduced model is selected using an asymmetric architecture that has input, four hidden layers and output layer with 512-1024-2048-4096-512-1 nodes. Total of 11,800,000 and 13,170,000 trainable pa-

rameters are used in the Full and Reduced models, respectively. The optimal selected *Activation (hidden)* function for the input and hidden layers is '*sigmoid*' for the Full model and is '*relu*' for the Reduced model. Other hyperparameters for both Full and Reduced NN models are selected as: *Batch size*=32, *Learning rate*=0.0011 and *Optimizer*='Adam'. No significant difference is found using different kernel regularizers and weight initializers. Therefore no regularization was used, and weights were set using default *glorot\_uniform* initializer. Other parameters are fixed to *Epochs*=200 and *Activation (output)*='linear'.

When dealing with high-dimensional limited sample data, Network models lead to overfitting and model instability due to highly variant gradients (102). Hence, identifying the optimal Reduced model architecture requires more complexity and includes more trainable parameters than the Full model during the optimization step. Such behavior can be justified by that the network models tend to compensate through the generation of new internal features to capture hidden non-linearity in the Reduced sparse setting. Unlike SVR, the number of parameters for NN is fixed and does not increase with the addition of new noisy data. As a result, computational time increases marginally, which can be observed in average runtime in Table 2.7. After testing the final model in the Real-time scenario, we observe that the model performs worse than the Full model. Therefore, hyperparameter optimization is biased towards reducing the validation error that follows the training set distribution. Hence, Bayesian optimization does not find the optimal model capable of generalizing on unobserved data.

Since we obtain two distinct architectures through BO, referred to as *Shallow* for Full model and *Deep* for Reduced model, we now cross-check their overall generalization ability. To this end, we employ heuristics on the constructed models, i.e., Shallow and Deep architectures for both Full and Reduced settings for

the Real-time scenario (Table 2.6). As determined through optimization for the Reduced setting, the Shallow model performs better on unseen data than the Deep model for the Full setting. In contrast, through an iterative reduction in the number of trainable parameters, based on the training (Figure 2.1a), validation (Figure 2.1b) and testing (Figure 2.1c) error, we observe that a less complex Shallow model (with only 0.5 million parameters) performed better in predicting unseen data observations compared to previously found “optimal” Reduced model with Deep architecture (Figure 2.2). Such scaled model is also computationally more efficient (Figure 2.3b). Hence, a more simple Shallow architecture proves to have better generalization ability for both Full and Reduced settings. Fine-tuned Reduced model with one hidden layer (472-954-1 nodes) is selected as the new optimal architecture.

	Full Model	Reduced Model
Shallow NN	32.3	42.5
Deep NN	34.2	55.8

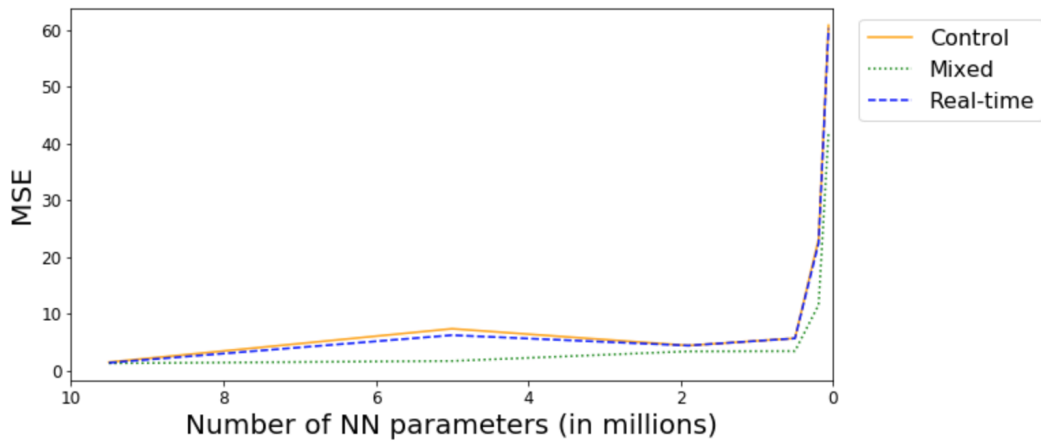
Table 2.6: NN architecture generalization ability on Real-time data, recorded as Test MSE using different architectures on Full and Reduced sets of data

### 2.3.4 | Feature Selection for Spectra Data

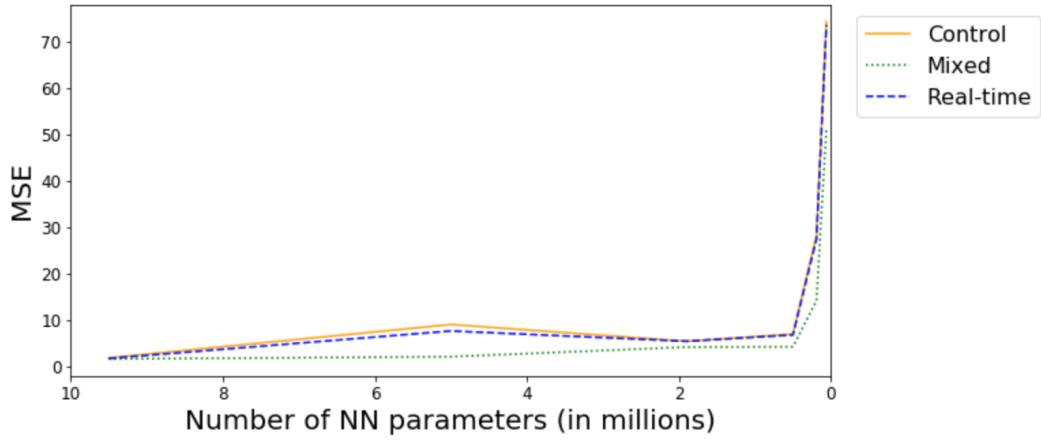
The following steps and tuning are performed to employ the considered feature selection and explainable AI techniques discussed in Section § 2.1.1 for spectroscopy analysis.

First, we select the optimal number of latent components for PCA and PLS by iteratively fitting spectroscopy data in scikit-learn corresponding decomposition libraries. For PCA, the number of components can be determined by plotting cumulative explained variance for  $p$  components and selecting optimal number using point of maximum curvature (Figure 2.4). Since PLS considers the

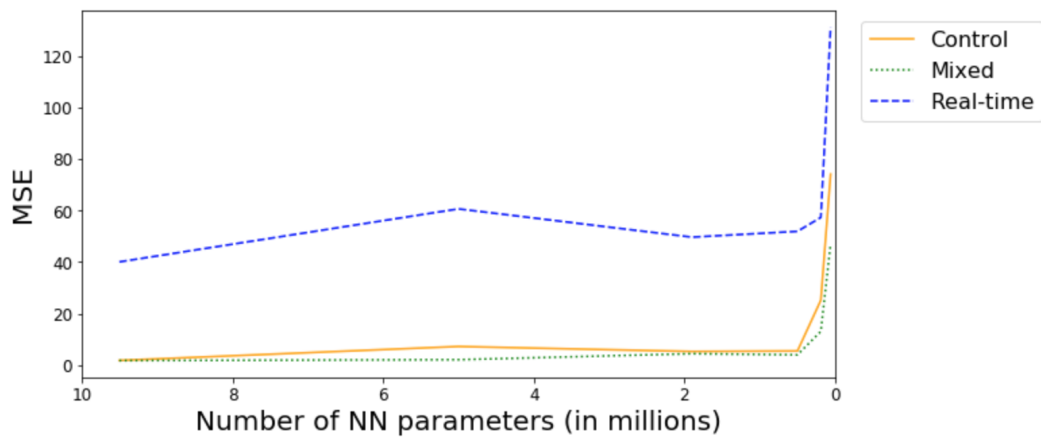




(a) Train MSE



(b) Validation MSE



(c) Test MSE

Figure 2.1: Average (a) Training, (b) Validation and (c) Testing error of Shallow NN on reduced subsets for various number of trainable parameters in three scenarios

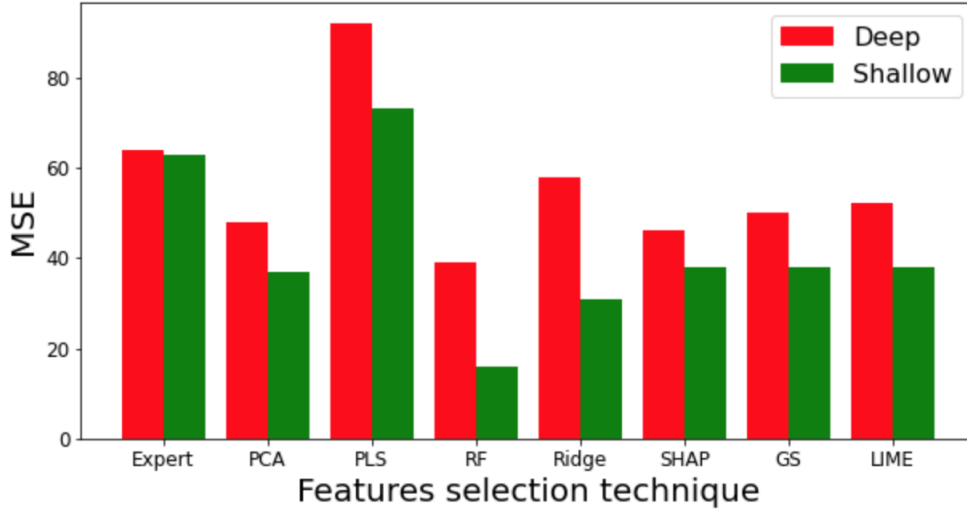
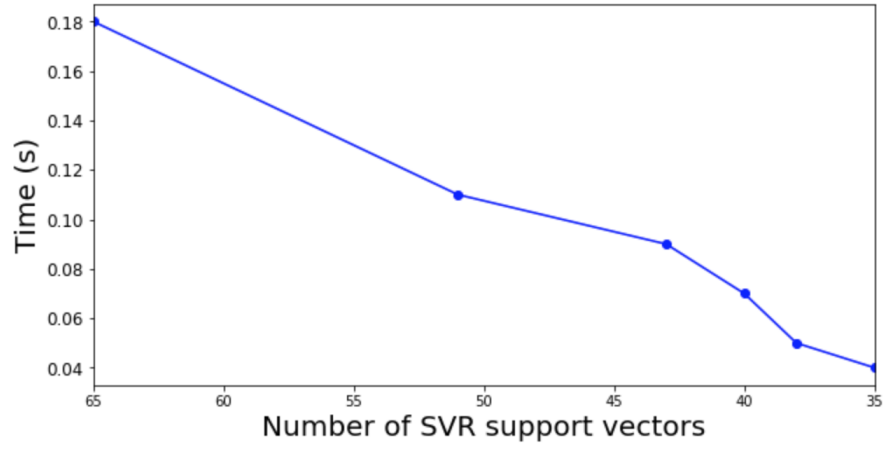


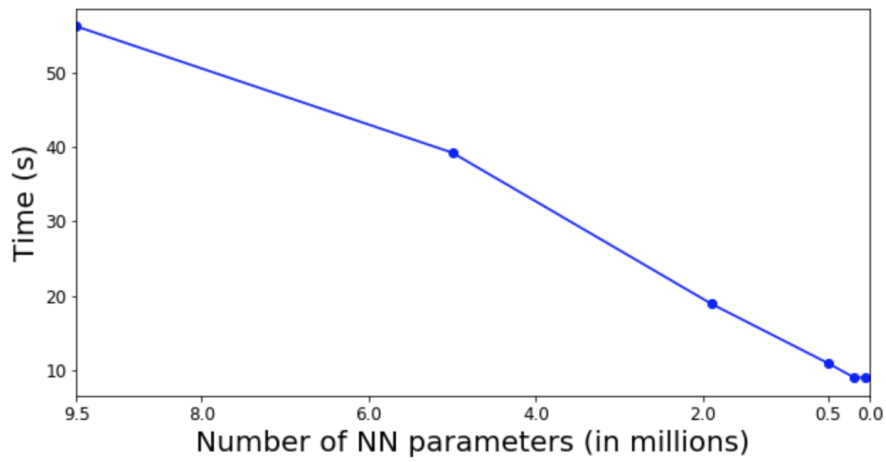
Figure 2.2: Shallow NN vs DNN testing error on reduced subsets in Real-time scenario.

relationship between feature space and predicted output, we can determine an optimal number of components based on the uncertainty of test results for the different numbers of latent variables. Therefore, using PLS Regression, we fit the cross-decomposed data and record Mean Square Error (MSE) between predicted and true values of CN for a given number of components (Figure 2.5). Hence, the optimal number of latent components is selected based on the lowest associated MSE value. The optimal number of PCA components is determined to be  $P = 4$ , and the optimal number of latent dimensions for PLS is determined to be  $P = 7$ .

Consequently, we employ one additional step to determine the true contribution of each feature to the overall explained variance within the data. To accomplish that, we first calculate Explained Variance (EV), which is the ratio between the variance of that principal component and the total variance in data for PCA or predictions for PLS. We then take each component and multiply its EV with the loading vector to obtain individual feature importance.



(a) SVR



(b) NN

Figure 2.3: Average computation time of (a) SVR (b) NN on reduced subsets of data for Real-time scenario with various model complexity levels.

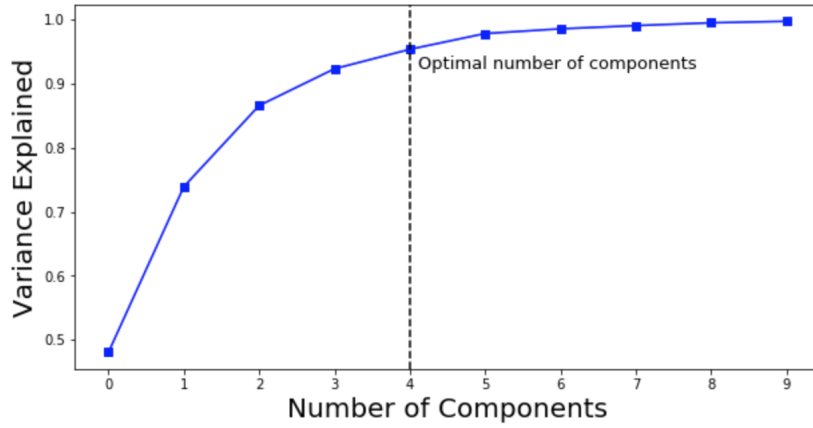


Figure 2.4: PCA cumulative sum of explained variance by number of components

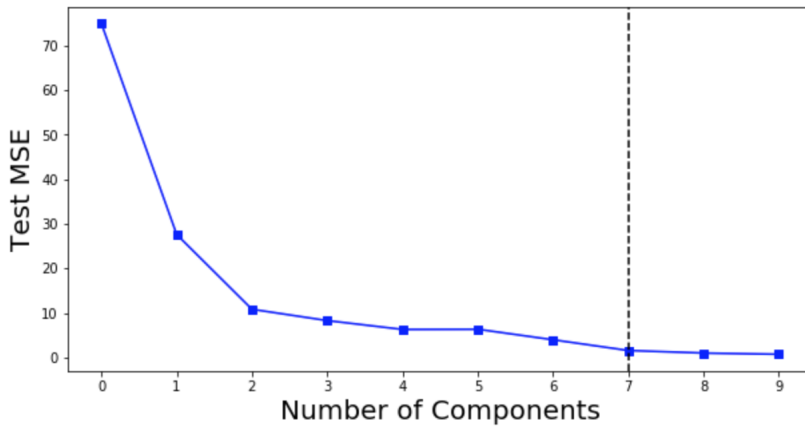


Figure 2.5: PLS Regression cross-decomposition and prediction results by number of components

$$EV_1 = 1 - \frac{MSE_1}{\sum_i^N (y_i - \bar{y})^2} \quad (2.23)$$

$$EV_{p+1} = EV_p - \left(1 - \frac{MSE_{p+1}}{\sum_i^N (y_i - \bar{y})^2}\right), \quad (2.24)$$

The *sklearn* toolbox allows us to directly access both the explained variance by component and loading vectors for PCA, while for PLS, we calculate EV of the first component  $p = 1$  using formula (2.23). For the remaining components

$p = 2, \dots, P$ , the EV is calculated using Equation (2.24) to return the final cumulative explained variance by component normalized between 0 and 1. The features with the highest absolute weight values across all components are then aggregated and sorted. After filtering out features with zero weight, we achieve a true collection of selected features and their corresponding wavelengths that help explain most of the variance in the data.

To determine the optimal number of trees ( $T$ ) to be used in the Random Forest Regression model and select the optimal  $\alpha$  parameter for the Ridge Regression model, we use a similar approach as for PLS. We iteratively fit the data to the above-mentioned models and record MSE between predicted and true values of CN for given parameter  $T$  in the range  $[0.001, 1]$  and  $\alpha$  in the range  $[1, 200]$ . The optimal number of trees for the Random Forest Regression model is  $T = 91$ . Similarly, for Ridge Regression we find optimal parameter value  $\alpha = 0.001$ .

After the adjustments, 299 features are found to explain over 80% of the total variance in training data using PCA, and 604 features are necessary using PLS, which is twice more than PCA selects. By extracting feature importance ( $\mathbf{w}$ ) from the Random Forest model, we observe that 119 features account for 80% of explained variance in data, significantly less than both PCA and PLS. As for Ridge Regression, 175 features with the highest corresponding linear weights ( $\mathbf{w}$ ) explain most of the variance in the data.

We use the LimeTabular package (103) to locally explain the behavior of optimized and trained NN and SVR models using Linear Regression as a simple model approximation. After perturbing the interpretable model input, features that contribute to individual fuel sample CN prediction are calculated, one observation at a time. In order to obtain a global explanation across the entire distribution of data, we record features with the highest associated coefficients (weights  $w$ ) for each individual sample explanation using LR as a surrogate

model. We then rank features selected most commonly across all sample instances and select them as the final subset of the most important features.

The SHAP explainer (104) takes any combination of a predictive model and masker, which constrains the rules of the cooperative game used to explain the model and returns a callable subclass object that implements an estimation algorithm. We use SVR and NN prediction functions as input models to be explained and select a random subset of samples, generated by random seed function, to be used as a background (masking) set. By stratifying the background set on the model output results, instead of using the whole training data, we drastically reduce calculation time while maintaining a good representation of our sample distribution. For Global Surrogate (105), similar to LIME, we use an interpretable linear regression model to estimate general model behavior.

### 2.3.5 | Discussion

Figure 2.6 illustrates the results of the considered feature selection techniques on our Control scenario dataset. The red bars indicate the selected features by the feature selection methods and the yellow bar indicates the features selected by the domain expert. As we can observe, the results indicate partial overlap of selected features with expert features around “fingerprint” region ( $500\text{-}1800\text{ cm}^{-1}$ ) and “Carbon-Hydrogen (CH) stretching” ( $2800\text{-}3200\text{ cm}^{-1}$ ) region, depending on the feature selection technique. Visual inspection suggests that similar subset of features is selected with model-agnostic methods. Note that features selected by Random Forest are notably more uniformly spread across the entire spectrum than other approaches and have the least number of features from the CH stretching region. This behavior can be justified by the fact that Random Forest considers different subsets of features randomly selected for each of the

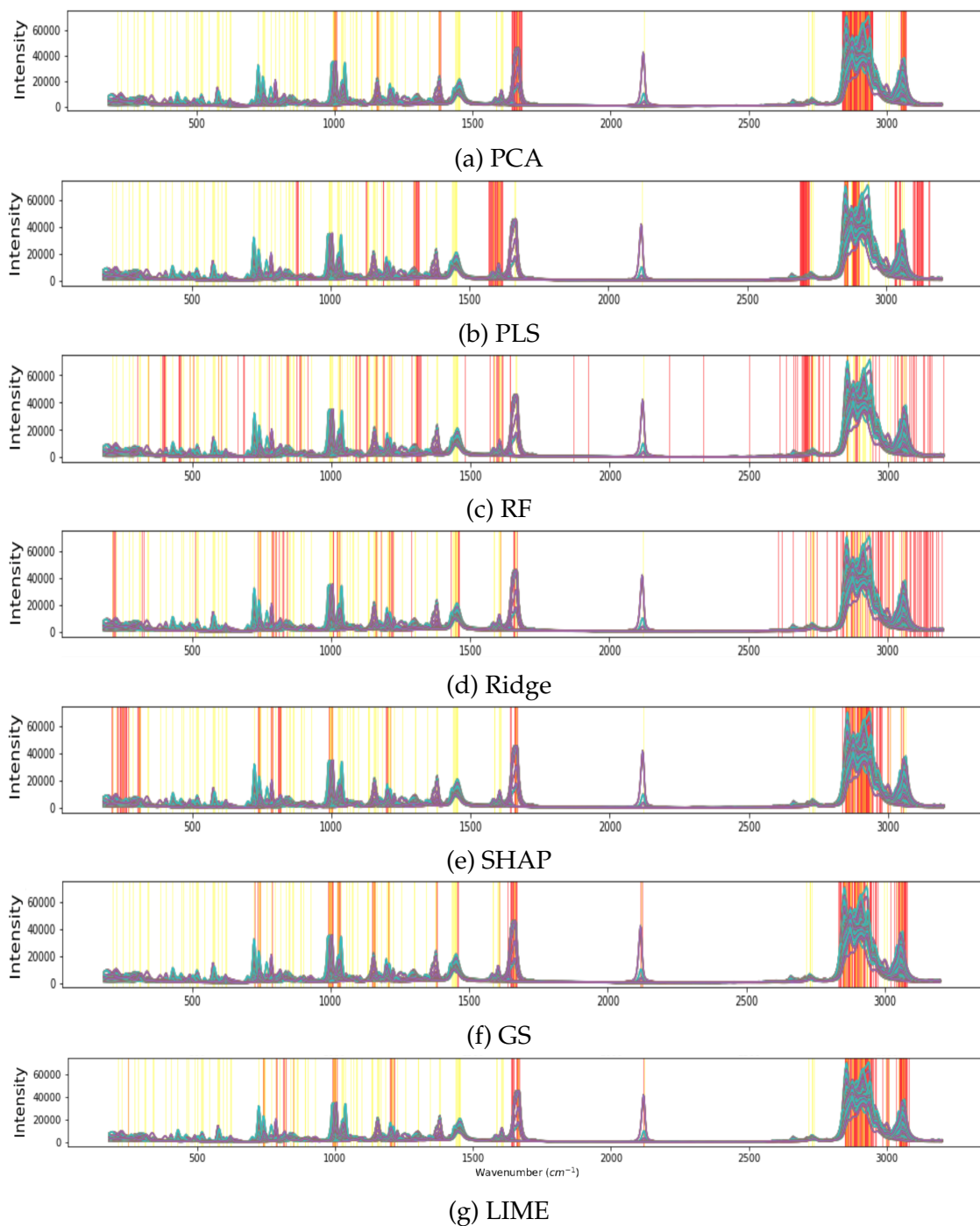


Figure 2.6: Regions of importance selected by various techniques. (a) PCA, (b) PLS, (c) Random Forest, (d) Ridge, (e) SHAPley, (f) Global Surrogate, (g) LIME are highlighted with *red* and expert features are highlighted with *yellow*.

individual models (Decision Trees) in its bag of models. Such considerations allows different attributes to be shown up in the tree structures and be considered for model construction. As a result, we can observe a wider range of locations selected as important in the spectra data. PLS and PCA results are more clustered in different regions. Both of these techniques are based on components that are linear combinations of original features. Hence, if attributes in one specific region are all important, they are assigned large loading values in multiple components and ranked higher in the final selected attributes.

Figure 2.7 represents the correctness evaluation of the considered techniques. The plot shows similarity proportion between subset of 120 Expert features  $S_{Ex}$  and 120-500 features selected using various techniques  $S_F$ . We immediately observe that PCA and PLS are outperformed by other techniques in correlating to expert features even when we increase the upper bound on the number of selected features, averaging 17%-60% from 120 to 500 features. The results indicate that LIME, SHAP, Ridge, GS, and Random Forest feature selections performed competitively in model explainability resulting on average in 33%-85% of overlap with the selected features by the expert going from 120 to 500 features. Further, we notice that selecting 120 features from each method only partially overlaps with features selected by the expert. Recall that the exact locations of the majority of 120 expert-selected wavenumbers are rough estimations by the domain practitioners. Further, only a small subset of these features contribute to the variance in CN prediction. It is worth mentioning that the features selected by the considered feature selection methodologies are mainly returned, optimizing prediction performance. Hence, there are other locations than  $S_{Ex}$  selected by such techniques. Since the expert selected features are based on rough peak location approximations, there might be actual locations that are missed. ML-based feature selection techniques might actually be able to discover the true



locations of the functional groups' peaks. However, there is no solid approach to confirm this statement.

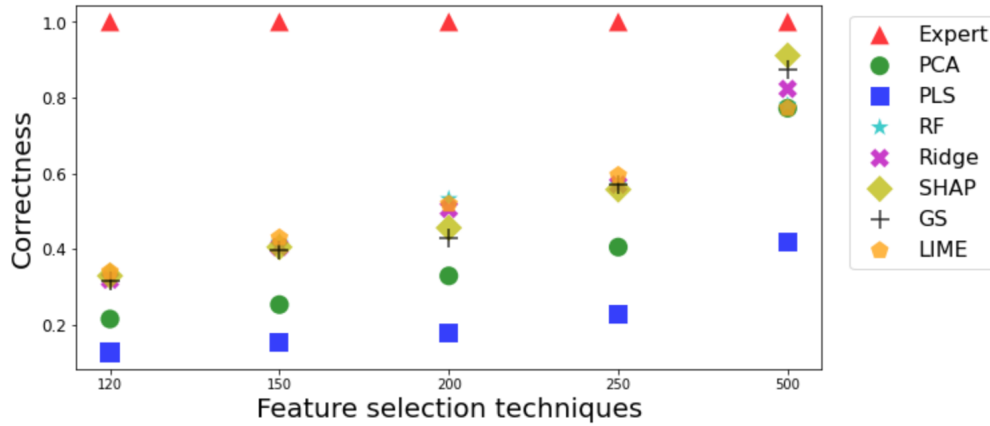


Figure 2.7: Method Correctness, displayed as proportion of wavenumbers matched with the domain-defined features.

The performance is measured using mean training and testing error ( $\mu$ ) with corresponding mean standard deviation ( $\sigma$ ) across 30 random splits of data. The model is considered to have better prediction performance when testing MSE is low and has better stability when deviation values are low as well. As can be seen in Table 2.7, the NN model tuned through the Bayesian Optimization approach (BO-Tuned NN) initially results in low testing MSE for Control and Mixed scenarios for both Full and Reduced sets of features. In a Real-time scenario, however, the minimum MSE is found to be 32.3 and 39.5 for Full and Reduced sets, accordingly. In contrast, after fine-tuning the NN, we can observe an increase in testing MSE for both Control and Mixed scenarios, while the error for Real-time scenarios becomes significantly smaller, achieving the new minimum MSE of 16 in the Reduced setting using Random Forest. While not included in the table, using Linear Regression, we achieve the lowest MSE of 88.8 for the Full model and 233 for the Reduced model, using the Random Forest subset of attributes.

Using the BO-Tuned SVR model, we initially obtained more competitive testing results across all three scenarios, achieving the lowest MSE of 33.7 for the Full set and even lower 32.9 for the Reduced set of features. After adjusting the SVR margin parameter, the Fine-Tuned SVR model achieves an all-time low MSE of 14.8 for Full and 4.6 for the Reduced set of attributes, also selected by Random Forest. Figure 2.8 summarizes numerical prediction results and associated deviation of the Fine-Tuned Support Vector and Neural Network regression models for the Real-time scenario for ease of comparison between two models.

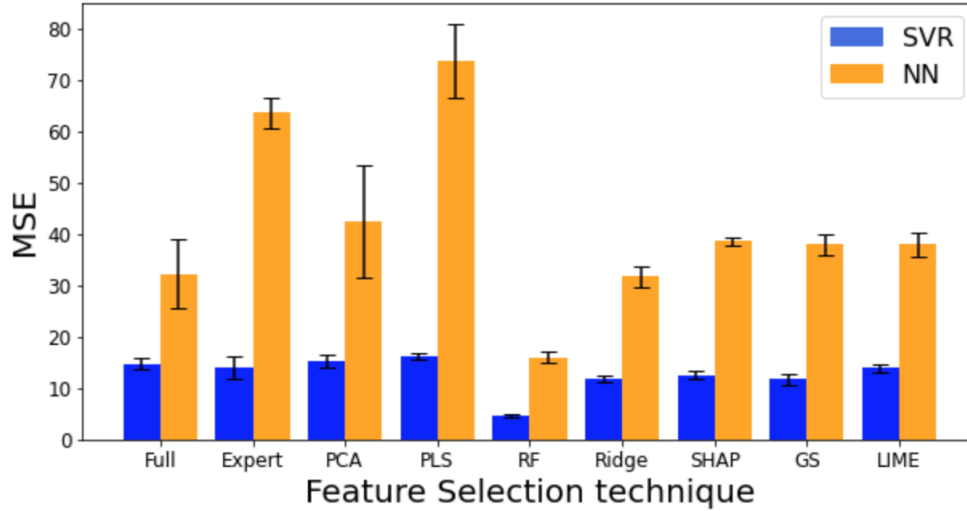


Figure 2.8: Testing error for Fine-Tuned SVR and NN models on Full and Reduced subsets in Real-time scenario.

The “curse of dimensionality”(106) affects both models equally, therefore only through a careful selection of hyperparameters we are able to avoid excessive overfitting. Although we reach similar performance for NN in some cases compared to SVR, parameter optimization for SVR, in general, is a much easier task, as fewer parameters are considered. As a result, after thorough tuning, Fine-Tuned SVR performs significantly better across all three scenarios. As mentioned above, the Reduced Fine-Tuned SVR and NN models trained on features

selected by Random Forest have significantly lower testing error and variance in Real-time scenarios compared to the Full model and is, therefore, selected as the best performing feature selection technique. We can explain the superior performance of RF feature selection in that it creates decision trees on boot-strapped subsets of observed data and randomly selected subsets of features (as discussed in Section § 2.2.3.1) before taking the majority vote on features that reduce the variance. Since this ensemble model considers different combinations of data and features, it has a better ability to identify important attributes for prediction. Since we are dealing with a small number of samples, one or two observations can really change overall data distribution. Therefore, RF has a better sense of variance when translated to unobserved data and results in better generalization ability.

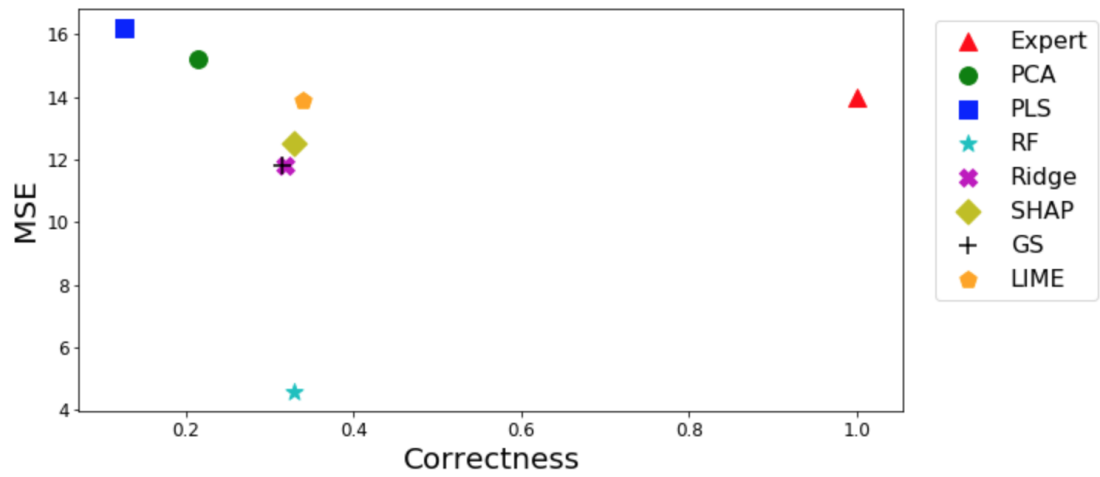
Using the Fine-Tuned SVR model as a benchmark, we observe that reduced models trained on features selected by Ridge, SHAP, GS, and LIME have competitive testing error with that of expert-selected features. One explanation, as mentioned earlier, is that not all expert features affect the CN prediction, while features selected with the above-mentioned techniques actually do.

PCA and PLS show the most unexpected results since these revered methods perform noticeably worse under different settings in both correctness and performance categories compared to all other methods. One possible explanation for their previous widespread use in chemometric analysis is the assumption that most chemical processes and reactions are linear. Both PCA and PLS decompose the data into linear combinations of original features and are commonly paired with linear regression in the literature.

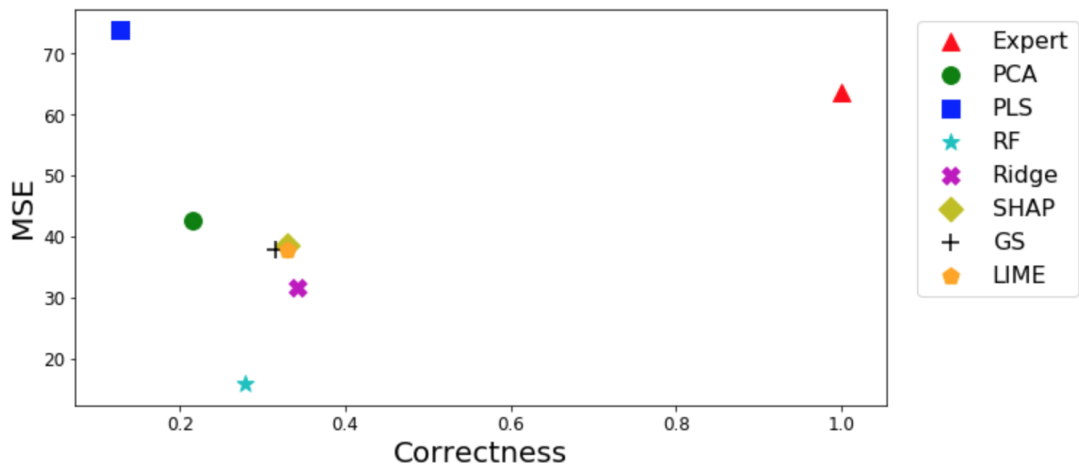
However, given the non-linearity of spectroscopic data, presence of statistical noise, and overall high dimensionality, we argue and attempt to support with our data that the particular relationship between fuel spectroscopy data and its

associated Cetane Number is not linear. Therefore, further attempt at selecting a reduced number of important attributes using these methods results in their poor performance. Additionally, PCA and PLS have simple and efficient implementation. As a result, advanced methods that can be superior to PCA and PLS in identifying the importance of spectroscopic features appear to have been overlooked in the literature thus far. While PCA and PLS analytical methods have been effectively used in many applications even outside of chemometrics, they prove to be the least suitable feature extraction method for our scope of work when compared to other methodologies.

Figure 2.9 demonstrates the Correctness versus Performance trade-off for both Fine-Tuned SVR and NN models, we can conclude that Random Forest selects the most meaningful subsets of attributes both from an explainability and prediction performance standpoint. This paper shows that optimizing hyperparameters for NN can be a challenging task that requires paying significant attention to the underlying data distribution. We also determine that given the high-dimensional limited size of data, SVR predictive model outperforms NN, although a further increase in the size of data can tip the scales back in favor of NN.



(a) SVR



(b) NN

Figure 2.9: Testing Error on Real-time dataset vs Correctness trade-off for 120 selected features using (a) SVR and (b) NN models

Number of samples	145			245			165			
FS method	Control			Mixed			Real-time			
	Train ( $\mu, \sigma$ )	Test ( $\mu, \sigma$ )	Time (s)	Train ( $\mu, \sigma$ )	Test ( $\mu, \sigma$ )	Time (s)	Train ( $\mu, \sigma$ )	Test ( $\mu, \sigma$ )	Time (s)	# parameters
Full Model	(0.006, 0.004)	(0.04, 0.04)	2338	(0.02, 0.05)	(0.07, 0.13)	2404	(0.01, 0.01)	(32.3, 6.7)	2533	11.8 mil
Expert	(0.02, 0.006)	(0.04, 0.02)	3506	(0.02, 0.03)	(0.03, 0.15)	4170	(0.03, 0.02)	(64, 19)	2922	13.2 mil
PCA	(0.05, 0.02)	(0.07, 0.03)	3607	(0.08, 0.02)	(0.17, 0.06)	4230	(0.04, 0.02)	(48, 20)	2439	13.2 mil
PLS	(0.29, 0.18)	(0.5, 0.58)	3896	(2.07, 0.85)	(1.6, 1)	4832	(1.66, 0.7)	(92, 17.2)	2835	13.2 mil
RF	(0.05, 0.01)	(0.08, 0.03)	3421	(0.04, 0.02)	(0.06, 0.02)	4260	(0.09, 0.08)	(39.5, 12.8)	2421	13.2 mil
Ridge	(0.05, 0.03)	(0.07, 0.03)	3327	(0.05, 0.01)	(0.07, 0.03)	4122	(0.06, 0.07)	(58.5, 20.6)	2855	13.2 mil
SHAP	(0.04, 0.02)	(0.05, 0.02)	3266	(0.02, 0.01)	(0.02, 0.01)	4620	(0.04, 0.025)	(46, 32)	2806	13.2 mil
GS	(0.04, 0.01)	(0.06, 0.03)	3285	(0.04, 0.02)	(0.06, 0.02)	4410	(0.03, 0.027)	(50, 12)	2460	13.2 mil
LIME	(0.03, 0.01)	(0.05, 0.03)	3217	(0.05, 0.02)	(0.09, 0.03)	4350	(0.06, 0.035)	(52, 14)	2444	13.2 mil
Avg. runtime	111			139			88			
Full Model	(0.006, 0.004)	(0.04, 0.04)	2404	(0.024, 0.045)	(0.07, 0.127)	2617	(0.01, 0.01)	(32.3, 6.7)	2613	11.8 mil
Expert Features	(0.05, 0.03)	(0.25, 0.2)	344	(0.19, 0.35)	(0.26, 0.5)	375	(5, 1)	(63.7, 3)	351	0.5 mil
PCA	(1.3, 0.3)	(4.9, 2.5)	380	(0.5, 0.83)	(0.67, 1.1)	391	(0.21, 0.5)	(42.6, 11)	381	0.5 mil
PLS	(22, 15)	(19, 20)	391	(12.3, 1.9)	(13.9, 4.6)	462	(11.7, 0.68)	(73.8, 7.2)	366	0.5 mil
RF	(0.86, 1.15)	(1.62, 1.9)	377	(4.6, 0.8)	(5.7, 1.9)	358	(0.73, 1.32)	(16, 1)	456	0.5 mil
Ridge	(0.73, 0.96)	(2.7, 3.8)	361	(0.6, 0.96)	(1.03, 1.95)	377	(3, 0.34)	(31.7, 2)	396	0.5 mil
SHAP	(0.001, 0.0015)	(0.06, 0.13)	384	(0.67, 0.87)	(0.82, 0.996)	470	(3, 0.4)	(38.6, 0.9)	357	0.5 mil
GS	(0.001, 0.008)	(0.02, 0.029)	362	(0.83, 1.19)	(0.83, 1.07)	383	(0.35, 0.63)	(38, 2.1)	369	0.5 mil
LIME	(0.048, 0.07)	(0.225, 0.18)	366	(0.42, 0.72)	(0.5, 0.67)	376	(0.13, 0.43)	(38, 2.3)	333	0.5 mil
Avg. runtime	19.89			21.51			20.82			
Full Model	(0.006, 0.002)	(0.02, 0.008)	0.58	(0.007, 0.0002)	(0.05, 0.019)	11.4	(0.006, 0.002)	(33.67, 5.44)	0.63	
Expert Features	(0.006, 0.002)	(0.02, 0.009)	1.2	(0.01, 0.007)	(0.05, 0.02)	35.97	(0.006, 0.002)	(33.95, 6.4)	1.2	
PCA	(0.007, 0.002)	(0.04, 0.015)	2.75	(0.026, 0.04)	(0.1, 0.08)	42.25	(0.007, 0.002)	(34, 5.5)	2.65	
PLS	(0.016, 0.007)	(0.32, 0.115)	30.8	(1.3, 1.5)	(2.46, 2.3)	61.12	(0.016, 0.007)	(42.5, 9.2)	30.27	
RF	(0.008, 0.002)	(0.06, 0.03)	5.2	(0.8, 0.74)	(0.36, 0.22)	56.78	(0.008, 0.002)	(41.2, 2.1)	5.35	
Ridge	(0.006, 0.003)	(0.03, 0.013)	1.3	(0.07, 0.07)	(0.13, 0.08)	41.97	(0.007, 0.003)	(37.9, 9.1)	1.33	
SHAP	(0.006, 0.002)	(0.03, 0.01)	1.8	(0.11, 0.3)	(0.46, 1.83)	39.84	(0.006, 0.002)	(32.85, 4.7)	1.83	
GS	(0.006, 0.002)	(0.03, 0.013)	1.7	(0.14, 0.58)	(0.44, 1.67)	38.93	(0.007, 0.003)	(36, 6.3)	1.63	
LIME	(0.006, 0.002)	(0.02, 0.008)	1	(0.027, 0.06)	(0.08, 0.05)	37.63	(0.006, 0.002)	(36, 3.74)	1.06	
Avg. runtime	0.17			1.4			0.2			
Full Model	(0.07, 0.001)	(0.08, 0.02)	0.41	(0.007, 0.0002)	(0.11, 0.03)	5.18	(0.006, 0.002)	(14.8, 1.1)	0.45	
Expert Features	(0.06, 0.001)	(0.08, 0.02)	0.36	(0.06, 0.001)	(0.11, 0.04)	17.2	(0.006, 0.002)	(14, 2.1)	0.36	
PCA	(0.07, 0.003)	(0.11, 0.03)	0.97	(0.06, 0.002)	(0.1, 0.08)	24.1	(0.007, 0.002)	(15.2, 1.2)	0.98	
PLS	(0.07, 0.003)	(0.13, 0.04)	5.1	(1, 0.79)	(1.5, 0.9)	39.2	(0.016, 0.007)	(16.2, 1.4)	5	
RF	(0.06, 0.002)	(0.08, 0.03)	0.49	(0.06, 0.001)	(0.09, 0.02)	5.5	(0.008, 0.002)	(4.6, 0.2)	0.5	
Ridge	(0.06, 0.002)	(0.08, 0.02)	0.37	(0.06, 0.001)	(0.1, 0.03)	7.9	(0.007, 0.003)	(11.8, 0.7)	0.37	
SHAP	(0.07, 0.003)	(0.1, 0.03)	0.7	(0.06, 0.002)	(0.14, 0.05)	22.3	(0.006, 0.002)	(12.5, 0.7)	0.73	
GS	(0.06, 0.002)	(0.1, 0.02)	0.68	(0.06, 0.002)	(0.12, 0.04)	16.1	(0.007, 0.003)	(11.8, 1.1)	0.7	
LIME	(0.06, 0.002)	(0.1, 0.02)	0.56	(0.06, 0.002)	(0.12, 0.03)	14.7	(0.006, 0.002)	(13.9, 0.9)	0.55	
Avg. runtime	0.04			0.6			0.04			

Table 2.7: Performance of NN and SVR predictive models fitted with Full and Reduced sets of features.

# Synthetic Data Generation

## 3.1 | Background

The idea of generating synthetic representations of real-time data using algorithms has a long history. Artificial data generation processes has been widely applied in many fields including perception (107; 108; 109) and speech synthesis for natural language processing (110; 111), where large volume of samples is required to train image recognition and speech models.

Before introducing deep generative models capable of generating unique new samples, the common approach to increase the size of an existing dataset was data augmentation. This technique is used to create slightly modified copies of existing data and serves the role of regularizer to reduce overfitting when training a predictive model. Popular data augmentation strategies include inverting, scaling, shifting or simply adding statistical noise to existing data to produce new samples. Data augmentation is a viable solution for many ML modeling domains, particularly image processing, since visual transformations only affect the context in which the item is displayed, not the object itself. However, permuting primary data might result in cardinal changes and the loss of key information in a new sample. As a result, it is indeed critical to create new

samples that retain the knowledge about the underlying sample intact. Hence, a reliable domain-agnostic technique for synthesizing data called generative modeling is introduced.

Generating realistic data that meet certain conditions absent in real data to design new cases and simulate new outcomes is well-motivated. Synthetic data can also be used to represent real outcomes without disclosing information about real data and therefore jeopardizing its confidentiality and privacy. Up until the introduction of Autoencoder networks (112), the term generative model broadly meant a class of statistical models that learned a joint probability distribution of observable and target variables. Simple generative models memorize the given input data distribution and have the ability to reproduce new samples that come from such distribution. However, the advance in neural network processing opened a new pathway for deep unsupervised representation learning. The two most popular deep generative methods currently are Autoencoders, and Generative Adversarial Networks (GANs) (113). In Autoencoder, by converting high-dimensional data to lower-dimensional data, an encoder model is forced to learn a compressed knowledge representation of the original data. This transformation is accompanied by trading off between two parameters: first, reconstruction accuracy, which measures the differences between original input  $x$  and the reconstructed  $\hat{x}$  space  $\mathcal{L}(x, \hat{x})$ , second, regularization term, i.e., L1 or Kullback–Leibler divergence, that discourages memorization or overfitting of training data. The compressed representation, therefore, learns to reproduce the input without holding onto its redundancies, allowing the production of diverse samples different from the original ones yet faithful to their underlying properties. Gradient descent is typically used for fine-tuning such network weights.

Alternatively, the approach of using adversarial training focuses on producing samples that are more realistic, rather than memorized with the greatest



accuracy, resulting in a more diverse output. Both techniques have their pros and cons. While GAN currently produces higher fidelity samples than Autoencoder networks, it is also difficult to optimize due to unstable training dynamics. However, since our goal is to produce the highest possible quality samples to improve data representation, all the while adhering to domain explainability constraint, we chose GAN approach for synthetic data modeling and discuss its details further.

### 3.1.1 | Related Works

Since its inception, GAN has been predominantly used in text, image, and video processing applications, generating near-perfect synthetic text (114) and images that are often indistinguishable from real by the human eye (115). Particularly for image processing, this ability to capture knowledge about given objects and interpolate it to new, unique object space was soon used to create image-to-image models capable of reconstructing and colorizing images (108; 116), blending different images together (117), improving image resolution quality (118), and create a whole spectrum of another photo/video editing applications. Many works are also dedicated to employing GAN for engineering design applications and are able to successfully generate chemical (119) and complex materials(120), compact electric circuits (121), and many other novel objects and designs. At this point, the application of GAN spans many domains, including but not limited to medical record synthesis (122), anomaly detection systems in cyber security (123), business, and data privacy-preserving (124), and many more.

While many classic generative and data augmentation techniques still are widely used in various applications, many fall short in robustness of GAN and its ability to deliver diverse, quality outputs. That being said, we review most

popular and recent literature on the use of data augmentation, encoding networks and GAN models for generating spectroscopy data.

The spectroscopic data augmentation process has been extensively investigated in (125) on a vegetable oils dataset for training a classification model to distinguish between spectra of 6 different vegetable oil species classes. The data augmentation procedure consisted of creating artificial samples by blending the weighted sum of input spectra from other samples, offsetting absorbance values on the spectral range axis, and adding white noise to observed values. The authors reported slightly improved classification accuracy, particularly through the use of samples generated with added statistical noise. The noise addition methods included uniform Gaussian noise, intensity dependant noise, and log-normal noise multiplication. It is also observed that only a small level of statistical noise injection leads to classification improvement before the performance drops off. Overall, data augmentation does not lead to a massive performance gain, so we turn our attention to deep generative techniques.

The application of Autoencoders for spectroscopic data synthesis was presented by (126) and was demonstrated to outperform state-of-the-art synthesis approaches at the time. Generating compressed representations of Raman spectroscopy via an encoding network allowed authors to produce new samples and yield improved classification accuracy for binary classification of samples that contain chlorinated solvents. The approach also incorporates the use of a trained autoencoder as an outlier detector which results in a model that both “produces high classification accuracy and is robust in the presence of negative outliers”.

Several papers have been released since 2020 that demonstrate how to use GAN to generate spectral data in order to improve baseline prediction model accuracy. In (127) 480 infra-red spectra of cumin and fennel plants were collected and used for classification using Quadratic Discriminant Analysis (QDA),

Multi-layer Perceptron network, and Convolutional Neural Network (CNN). The training data were decomposed using Principal Component Analysis (PCA) prior to fitting the model. The GAN model was used to expand the training set and improve classification performance by generating new samples. Each subset of plant samples was replicated using a separate GAN model, providing a narrow representation of real sample distribution, even within given classes of cumin or fennel. The classification accuracy results suggest that using GAN was not necessary in the first place, as the model accuracy was already high due to low problem complexity.

In (128) authors use Bidirectional GAN (129) to improve the result of CNN prediction model used for drug identification. In this work, 1721 samples of four drugs from 29 manufacturers were used to train the generative network and test the classification model. The motivation to use GAN was to solve the problem of insufficient samples necessary for efficient neural network training and unbalanced representation of all drug classes in terms of their training data subset size. The researchers collected FTIR spectral data and implemented an additional condition for the generator to generate the data for a given class. This was done by including an additional loss parameter that calculates classification error between synthetic data label and training data label. The authors discussed that such conditional GAN enhanced classifier performance over other models considered when at least 50% of data was used for training GAN. Although hyperparameter selection was mentioned, it was unclear how the model was impacted. The uniformity of generated samples used to balance the size of underrepresented subsets of drug classes suggests that such a model could suffer from a mode collapse.

In (130) GAN was used to expand the training set for laser-induced breakdown spectroscopy (LIBS) classification problem. Vanilla GAN was used to gen-

erate new samples that were normalized between 0 and 1. After synthesizing the data, the synthetic and real samples were compared using unsupervised PCA and K-means clustering techniques. The first two principal components and clusters of each spectra type were plotted against each other for visual inspection to compare their similarity. The percentage of generated spectra that was classified into experimental spectra was used as an estimation of given class synthetic data similarity with training data. The resulting sets of real and synthetic spectra were combined and a SVM classifier was fitted to estimate the change in classification performance. Correct classification rate (CCR) went from 88.89% to 95.33%, which indicated positive classification accuracy improvement.

In (131) the authors used hyperspectral image translated to normalized spectroscopic data to create additional samples to improve baseline classification performance for a limited size dataset. Two GAN architectures, namely Conditional GAN (CGAN) and Deep Convolutional GAN (DCGAN), were compared. Using CGAN, the image representations were conditioned with encoded label information, resulting in higher quality generated samples than using DCGAN architecture. The difference in quality was evaluated using a comparison of PCA components, training SVM and RF classifiers on the combined synthetic and real data, and observing changes in classifier recall and accuracy metrics. Although it was suggested that using this technique improves the baseline classifier's generalization ability, sufficient empirical or technical evaluation of this claim was present.

In another work by (132), the 100 spectra of three classes of marine pathogens were used to train separate GAN models to generate samples of a particular strain. The trained GAN discriminator was then used as a classifier model to evaluate pathogen classification accuracy using real and synthetic data. However, using a discriminator model in such a capacity is not meaningful, as it

is optimized for distinguishing between real and artificial samples rather than a multi-class classification task. No details on the GAN architecture or choice of hyperparameters were provided. Further, the difference in intensity values within generated data was used for region significance estimation. The authors argue that areas of highest variance are most significant for differentiating unique classes. Unfortunately, this claim is not further explored, leading to the belief that such variance is likely a result of statistical generated through the GAN training process.

A new approach was created using terahertz (THz) spectroscopy of rice and carbendazim powder tables to build 2-D heatmap "pictures" of such spectra, which were then used to train an image classifier to categorize samples depending on their pesticide residue content. The spectroscopy was transformed from a one-dimensional array to a two-dimensional picture  $I$  using the transpose of the THz spectrum absorbance coefficients as a function of  $I = x * x^T$ , where  $x$  is the sample absorbance vector (133). The classifier was then retrained using the ResNet network, and the authors argued that using transfer learning solves the issue of overfitting. The WGAN framework was used to ensure training convergence, and the generated samples were used for training SVM, KNN, and ensemble classifiers to compare their performance with CNN. The deep ResNet network scored slightly higher on differentiating between 13 classes than its reduced version. No solid evidence was provided that features extracted from training ResNet on 3.2 million images in 5247 categories of the ImageNet database reduced overfitting on unseen spectroscopic data. It can also be argued that such a deep network promoted overfitting and, therefore, yielding a higher testing accuracy on a given set.

Another interesting approach was discussed by (134), where GAN was used as a spectroscopy pre-processing tool instead of popular scatter-correction tech-

niques. Alternative to using scattering corrections to smooth the effects of measurement geometry and instrumentation noise, the proposed method involves using low-resolution spectra as latent statistical noise input in the generator portion of GAN, trained on high-quality spectra to process low-quality spectra into smoothed, processed ones. The results indicated that it is more computationally efficient to read raw spectroscopy in low resolution and process it with GAN rather than using long shutter collection time and standard pre-processing techniques. However, the real data with higher shutter stabilization yielded better classification results upon using such data for the training classification model. Another work that must be mentioned is published by (135) and builds on a previous work of (130) by introducing  $\gamma$  parameter to the WGAN network to control diversity and quality trade-off for generated data. The proposed parameter automatically controls the effort allocated to the training generator or discriminator module. However, no details on the implementation were presented.

After a thorough literature review, we noticed that using spectra blending and statistical noise injection to generate new samples via data augmentation yields marginal improvement. While encoding networks are shown to perform deep representation learning, GAN remains the most promising approach for spectroscopic data synthesis. However, given the preponderance of works published on this topic, the exact implementation, reproducibility of the results, and generalization still remain unclear. The majority of existing works fall short in providing clear details on technical solutions to address GAN training stability for spectra data, with many displaying model collapse behavior. Moreover, the methodology for evaluating the quality of the generated spectra is only limited to severe simple statistical analyses, with no discussion on how realistic it is or how reliable the data is. Therefore, we prioritize our work on developing a comprehensive technical guide to optimal spectroscopy GAN training and

investigating synthetic data quality from the domain standpoint and Machine Learning efficacy approach.

## 3.2 | Methodologies

### 3.2.1 | Generative Adversarial Network

GAN (113) is the most successful generative model developed in recent years, used in many applications from video editing and restoration to engineering design. Introduced in 2014 as a new way to generate data, this implicit density deep generative model aims to learn the true distribution of given data to generate new samples indistinguishable yet different from the original data. GAN is a hot topic of research currently, and there have been a variety of GAN implementations. The so-called “vanilla” GAN architecture is the simplest and most commonly used form of GAN. It is composed of two adversarial deep neural network models, generator ( $G$ ) and discriminator ( $D$ ), that compete in a min-max two-player game until near-Nash equilibrium is reached to generate data that plausibly follows the original data distribution. To this end, the generator ( $G$ ) first creates a mapping function from a supplied statistical noise distribution  $p_z(\mathbf{z})$  to a new data space that resembles training data space. The latent statistical noise vector  $\mathbf{z}$  is typically initialized by sampling values from Gaussian distribution with mean 0 and standard deviation of 1. The noise vector is later transformed into a synthetic sample once the underlying data distribution is learned. The role of discriminator ( $D$ ) is to output a variable that represents the probability that a sample ( $x$ ) is generated following the training data rather than a generator. GAN trains both models simultaneously, adjusting generator’s parameters to minimize  $\log(1 - D(G(\mathbf{z})))$  and adjusting discriminator’s param-

eters to minimize  $\log D(\mathbf{x})$ . This process can be represented mathematically as:

$$\min_G \max_D V(D, G) = E_{x \sim p(x)} [\log D(\mathbf{x})] + E_{z \sim p_z} [\log(1 - D(G(\mathbf{z})))] \quad (3.1)$$

where  $p$  is the real data distribution and  $p_z$  is the input latent noise vector. The goal of function  $G$  is to map from latent space to the data space, while function  $D$  differentiates between real and artificial samples.

Training a stable GAN is a challenging task as it is a highly hyperparameter-dependent model and involves improving both discriminator and generator models that work in tandem. Consequently, changes to one model can affect the performance of the other. Moreover, as discussed by (136), in practice, finding this Nash Equilibrium is a difficult task. Due to the non-convexity of  $D$  and  $G$ , there is no theoretical guarantee of obtaining the Nash Equilibrium. Hence, this optimization problem is usually solved using a gradient descent approach. The most common issue arising from training a GAN is called *mode collapse*. This happens when the generator begins repeatedly producing the same output (or a subset of outputs) while the discriminator fails to reject it. The task of the generator is to create plausible samples indistinguishable from real data to the discriminator, and the default strategy of the discriminator is to attempt to reject them. However, if, during one of the epochs, the discriminator gets stuck in a local minimum and cannot reject fake samples, the generator will then over-optimize for this particular configuration of generator weights and continue to generate the same samples that are proven to mislead it. Additionally, the respective loss functions of generator and discriminator models are not informative on their own, making tracking model convergence and hyperparameter optimization difficult, with most of the model tuning ending manually.



### 3.2.2 | Advanced GAN Architectures

Several GAN modifications have been proposed to enhance the model convergence and quality of data generated, such as Conditional GAN (CGAN) (137), Deep Convolutional GAN (DCGAN) (138), Wasserstein GAN (WGAN) (139), WGAN + Gradient Penalty (WGAN-GP) (140), Packing GAN (PacGAN) (141) and many others. Each approach utilizes particular subset of technical solutions that we refer to as modules. In this research, we compile a list of modules that can be utilized to train stable synthetic spectral data generators and attempt to determine their relative effectiveness through comprehensive ablation test. The following parameters and modules were considered:

*Latent Dimensions.* The latent dimension size determines the range of diversity of generated statistical noise. The generator takes the sampled vector and then maps it to the training data distribution by minimizing the Jensen-Shannon Divergence of the probability distribution of the sampled vector and the distribution of all the training data. Higher dimension value results in more variety of latent variables generated and slows down the convergence of generator training.

*Latent Noise Space.* The latent space defines the shape and distribution of the input to the generator model used to generate new samples. Most works advise sampling from a standard Gaussian distribution to initialize latent space. As such, the shape of the latent space is a hypersphere, with a mean of zero and a standard deviation of one.

*Network Capacity.* The model for the discriminator is usually more complex than the generator. Increasing generator capacity shows no improvement in the quality of the generated data if the discriminator is not providing quality feedback in the first place. Therefore, it is essential for the discriminator to be ahead of the generator.

*Activation Function.* Network activation functions are critical to construct a balanced, high-performing model with sparse activations. Activation sparsity is defined by the number of hidden units with the non-zero output after being activated, promoting more effective learning. The sigmoidal activation function is typically a good starting point for most research. However, it can also lead to saturation in both function directions, thereby resulting in a vanishing gradient problem. To remedy the vanishing gradient and promote sparse activations, a more advanced rectified linear activation unit (ReLU) function (142) was introduced in 2018 and is currently the most popular choice of activation function for deep networks. While most works recommend using the ReLU activation function for the generator (143), a variation of ReLU, called Leaky ReLU, is preferred to use in the discriminator for all layers. Leaky ReLU allows values less than zero by specifying its negative slope (default value is 0.2) and learns where the cut-off should be in each node of the network.

For classification and sampling tasks, such as differentiation between real and synthetic samples in discriminator or sampling from mixture of Gaussian distributions, as will be discussed later, we also use Softmax and Gumbel-Softmax activation functions. Since sampling discrete data from categorical distribution is not differentiable, a Gumbel-Softmax is used instead of Softmax to ensure that gradient estimation can be acquired. The Gumbel-Softmax function takes advantage of Gumbel distribution, which is a continuous distribution that approximates samples from a categorical distribution and enables the ability to backpropagate through samples. It has the essential property that it can be smoothly annealed into a categorical distribution (144) and therefore used for sampling discrete data.

*Batch Normalization.* Batch normalization standardizes the activations of a previous layer, which are added to the model after the hidden layer. Recentering

and rescaling the layer's inputs to have zero mean and unit variance produces smoother parameter space and stabilizes the training process.

*Dropout.* Introducing a dropout layer in the architecture would avoid the discriminator's overconfidence. By randomly dropping a subset of the layer's outputs in the discriminator, this method changes the number of nodes and connectivity of layers. Dropout makes the training process noisy, simulating the network to learn a sparse representation and is a better alternative to active regularization.

*Minibatch Discrimination.* To avoid a mode collapse of the generator, separate batches of real and synthetic data are used to train the discriminator. Hence, we discriminate between whole minibatches of samples rather than between individual samples. These minibatches are computed in the intermediate layer of the discriminator separately and are fed to the next layers together. Since the task of the discriminator is to classify individual examples as real data or generated data, it is able to simultaneously use labeled examples of both real and synthetic samples in the current batch of model inputs as reference information to make better decisions about whether new samples are real or not.

*Mode-specific Normalization.* Continuous value representation with arbitrary distribution is non-trivial. Based on an investigation conducted by (122), continuous values in tabular data are typically non-Gaussian, and min-max transformation result in vanishing gradient problems. Since spectroscopic data is stored as tabular data, with each individual column representing material absorbance or scattering intensity value for a certain spectrum wavenumber. Hence, the issue of appropriately pre-processing spectroscopic data are similar to that of tabular data. To deal with columns having non-trivial distributions, in other words columns whose values follow a mixture of multiple distributions, each column is treated independently using a mode-specific normalization. Therefore, given

data of size  $j \times i$ , where  $j$  represents a row and  $i$  a column index, a variational Gaussian Mixture Model (GMM) is applied for each column ( $C_i$ ) to estimate the number  $k$  of Gaussian distributions (which we will refer to as modes  $m$ ) present in that column. The equation for learned GMM is given in 3.3, where  $\mu$  and  $\sigma$  are mode weight and standard deviation accordingly.

For each value  $c_{ij}$  in column  $C_i$  we compute probability of it coming from particular mode, with probability densities given in equation 3.2. Once the mode is sampled, each value  $c_{ij}$  is normalized following equation 3.4. The mapping that stores data-specific GMM information is stored and used to transform newly generated data back to the original range.

$$\rho_k = \mu_k \mathcal{N}(c_{i,j}; m_k, \sigma_k), \quad (3.2)$$

$$\mathbb{P}_{C_i}(c_{i,j}) = \sum_1^k \mu_k \mathcal{N}(c_{i,j}; m_k, \sigma_k), \quad (3.3)$$

$$a_{i,j} = \frac{c_{i,j} - m_k}{4\sigma_k}, \quad (3.4)$$

where  $\mathbb{P}_{C_i}$  is a given  $C_i$  column's mixture model, represented as sum of  $k$  Normal distributions  $\rho_1 + \rho_2 \dots \rho_k$  with  $\mu$  and  $\sigma$  being the weight and standard deviation of respective mixture mode  $m$ , and where  $a_{i,j}$  is output vector of column values normalized by given mode  $m_k$ .

*Conditional Information.* The core principle of Conditional GAN is a method that extends the performance of vanilla-GAN models by conditioning both generator and discriminator on additional information ( $y$ ), which can be any “auxiliary” information such as class labels. The latent noise, together with  $\mathbf{y}$ , is combined in a joint hidden representation for the generator. In the discrimina-

tor,  $\mathbf{x}$  and  $\mathbf{y}$  are both used as inputs for a discriminative function. The updated Conditional GAN objective function then is as follows:

$$\min_G \max_D V(D, G) = E_{\mathbf{x} \sim p(\mathbf{x})} [\log D(\mathbf{x}|\mathbf{y})] + E_{\mathbf{z} \sim p_z} [\log(1 - D(G(\mathbf{z}|\mathbf{y})))] \quad (3.5)$$

where  $p(\mathbf{x})$  is the real data distribution and  $p_z$  is the input latent noise vector. The goal of function  $G$  is to map from latent space to the data space, while function  $D$  differentiates between real and artificial samples. In this configuration, both real data  $\mathbf{x}$  and latent noise  $\mathbf{z}$  are conditioned on some information  $\mathbf{y}$ .

*Wasserstein Loss.* Wasserstein GAN (WGAN) is one of the most popular GAN models and consists of an objective change that results in training stability and interpretability. As a result, WGAN is less sensitive to the choice of the Network architectures and hyperparameters. This approach challenges the classic GAN idea of learning the probability distribution of data by learning its probability density so that a family of densities is selected that maximizes the likelihood of the data. As the authors claim, to apply the likelihood maximization approach, such density must exist in the first place, which is not always the case when “dealing with distributions supported by low dimensional manifolds” (139). When the model manifold and the true distribution intersect, the divergence between them becomes undefined, leading to infinite values and mode collapse as a result. To counter this issue, a new method of determining how close the model and real distributions are is presented. In the process of identifying various ways to define a divergence, they introduce the Wasserstein-1 distance loss function, which essentially replaces a probability of discriminator loss with a score that correlates with the quality of generated data. Following the description from (140), the Wasserstein (or Earth-Mover) loss function ( $W(q, p)$ ) is informally defined as “the minimum cost of transporting mass” used to transform the distribution  $q$  into the distribution  $p$  where the cost is mass times transport distance.

This method allows for continuous training of the critic (discriminator) without reaching saturation, avoiding the mode collapse that the usual GAN method is prone to. Using such a distance function, which has solid theoretical properties compared to the originally defined distance metric, induces a stronger topology by delaying the convergence for a sequence of distributions.

Furthermore, the discriminator's parameters are clipped to a certain compact space to enforce a Lipschitz constraint, under which all neural network transformations and pointwise nonlinearities are smooth Lipschitz functions. While clipping weights, is "clearly a terrible way to enforce a Lipschitz constraint" (140), the authors left it for further research in the initial paper. Additionally, the discriminator's parameters are updated more often than the generator's parameters throughout each iteration to ensure a robust discriminator (named critic in this configuration) model is used to speed up the convergence to optimal model equilibrium. The new objective function is described as follows:

$$\min_G \max_{D \in \mathcal{D}} \mathbb{E}_{x \sim \mathbb{P}_r} [D(x)] - \mathbb{E}_{\tilde{x} \sim \mathbb{P}_g} [D(\tilde{x})], \quad (3.6)$$

where  $\mathcal{D}$  is the set of 1-Lipschitz constraint and  $\mathbb{P}_g$  is the model distribution implicitly defined by  $\tilde{x} = G(z), z \sim p(z)$ . In this setting, using optimal critic, minimizing the value function with respect to the generator parameters minimizes  $W(\mathbb{P}_r, \mathbb{P}_g)$ .

*Gradient Penalty.* As mentioned by (139), using weight clipping was not optimal solution to enforce Lipschitz constraint. Therefore, an alternative approach of penalizing the norm of gradient of the discriminator (critic) model with respect to its input is proposed (140). This method uses soft version of the 1-Lipschitz constraint with a penalty on the gradient norm for random samples  $\hat{x} \sim \mathbb{P}_{\hat{x}}$ . The following penalty component is added to (3.6):

$$\lambda \mathbb{E}_{\hat{x} \sim \mathbb{P}_{\hat{x}}} [(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2], \quad (3.7)$$

where  $\lambda$  is a penalty coefficient and  $\mathbb{P}_{\hat{x}}$  is uniform sampling the lines between pairs data distribution  $\mathbb{P}_{\hat{r}}$  and the generator distribution  $\mathbb{P}_{\hat{g}}$  points. It has been shown that this method outperforms the standard WGAN and enables stable training of a wide variety of GAN architectures with almost no hyperparameter tuning (140).

*Packing.* PacGAN solution to problem of mode collapse was introduced by (141). The main idea of packing is to “modify discriminator to make decisions based on multiple samples from the same class together, either real or artificially generated”, by passing combined samples to generator and discriminator simultaneously. It was demonstrated via binary hypothesis testing that packing penalizes generators with mode collapse, favoring generator distributions with less mode collapse. This approach requires minimal architecture adjustment and is proven to aid in GAN training convergence.

### 3.2.3 | GAN Evaluation

As mentioned in § 3.2, training GANs is difficult as the loss curves of the generator and discriminator oscillate in a non-informative manner, with no clear indication of whether the model is improving. This is the result of the min-max zero-sum game nature of GAN, where both players can undo each other’s progress (145). While several evaluation metrics exist, there is no clear consensus regarding which metric is the most appropriate to evaluate the GAN training process, or the quality of the output sample (146). While there is a number of proven metrics dedicated to the evaluation of images (146), we chose the following three categories of methods to evaluate the quality of generated spectroscopic data:

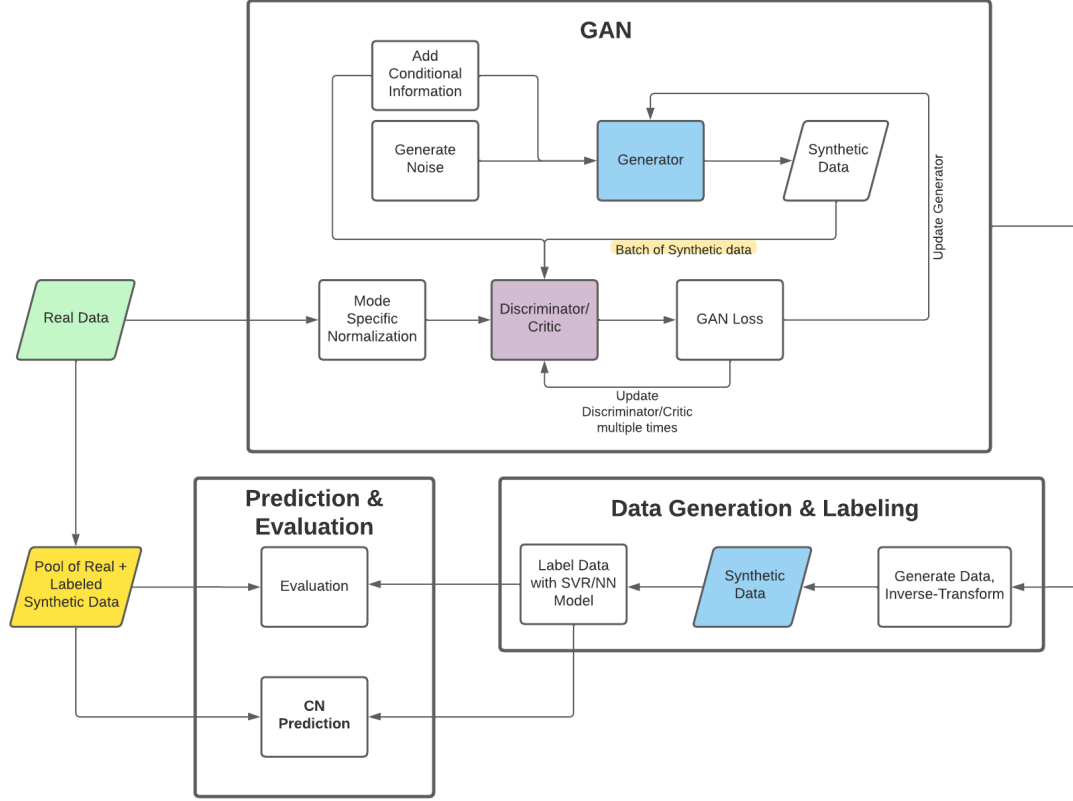


Figure 3.1: Complete synthetic data generation and evaluation process diagram.

Statistical Similarity, Model Efficacy, and Domain-expert Conformance. Figure 3.1 presents a complete overview of GAN training, sample generation, and evaluation process flow.

**Statistical Similarity.** Using Kolmogorov-Smirnov (KS) test, we can estimate the similarity between generated ( $S_{synth}$ ) and training ( $S_{real}$ ) data (147) as a function of the distance between cumulative distribution functions (CDFs) of the data they were respectively drawn from. The Kolmogorov-Smirnov statistic equation is:

$$D_n = \sup_x |F_n(x) - F(x)|, \quad (3.8)$$



where  $n$  is the number of samples,  $\sup_x$  is the supremum of the set of distances between values in each distribution. The statistic  $D$  therefore takes the largest difference between two distributions and converges to 0 when divergence is non-existent and, therefore, goes to infinity. Since spectroscopic data can be effectively described as tabular data where each column follows some underlying GMM, we apply a two-sided KS test on each column  $C$  between real and synthetic data samples and subtract 1 from the average of the results across all columns to get final KS score used for evaluation:

$$D_{avg} = \sum_1^i D_n, \quad (3.9)$$

$$KS_{score} = 1 - D_{avg}, \quad (3.10)$$

where final output  $KS_{score}$  indicates the maximum distance between the expected CDF and observed CDF values for two subsets of data. While a high KS score indicates that GAN successfully captured the underlying distribution of training data, an overly-high score might indicate a mode collapse of the generator. We use the  $KS_{score}$  to evaluate the quality of the generated data and overall GAN training stability.

As discussed in §2.2.2.1 principal component analysis (PCA) is commonly used to convert multiple indicators in high-dimensional data into a few representative lower-dimensional components (principal components (PC)), where each PC holds the significant amount of information about the original data. Various works take advantage of PCA simplicity and power to evaluate the prominent directions in GAN latent learning space (148) and estimate similarity between spectral data generated by GAN and real spectral data (131). In this research, PCA is used to project high-dimensional spectral data into low-

dimensional space to quantify the similarity between the spectra generated by GAN and the real spectra.

**Model Efficacy and Synthetic Data Labeling.** Machine Learning Efficacy can be used in addition to statistical analysis to evaluate the performance of using synthetic data as training data for predictive models (122). Using our predictive modeling framework discussed in Chapter §2.3.3, we train the best-suited prediction model on a synthetic dataset and evaluate prediction models using an independent test subset of the real data. While we can generate quality spectroscopic data, we also need an associated CN value for each sample to use this data in a supervised learning setting of a regression model. The “ground truth” CN values are commonly calculated using compressed fuel measured ignition delay obtained through Ignition Quality Tester (IQT) (149) or Cooperative Fuel Research (CFR) methods. However, since no real substance exists that is associated with generated spectroscopy, we devise a different strategy to acquire CN labels for our spectra. There is a recent branch in machine learning, referred to as Weak-supervision learning (150) that mainly deals with situations when large amounts of unlabeled data are available for a supervised learning setting. The core idea of this approach is that weak, noisy signals can be used to supervise the labeling process of such data and ultimately help create a strong predictive model. The assumption is that using data with “weak” (lower-quality) labels while understanding that they are imperfect ultimately leads to predictive model performance compared to not utilizing unlabeled data at all. Using this approach, we use a machine learning model trained on the existing labeled dataset to assign weak (CN) labels to newly generated samples. While it is expected that using imperfect labeling (oracle) model results in such weak labels, we expect to enhance model stability and overall model generalization to unforeseen data by expanding the diversity of the labeled dataset. As discussed

in §2.2.5.2, Mean Squared Error (MSE) is used as a performance metric to evaluate the accuracy of the regression model trained on synthetic data  $MSE_{synth}$  and compare with that of the baseline model trained on real dataset  $MSE_{base}$ . To ensure the reported results are robust and accurate, we take average MSE across 30 observations, generating new samples and relabeling them with randomly split real data every time. Lower quality of generated samples should result in lower prediction accuracy compared to baseline. In contrast, if the MSE values are equally similar, the synthetic data can be judged of acceptable quality.

**Domain-expert Conformance.** Domain-expert evaluation serves as a final evaluation step to ensure the quality of synthesized data. While the generated data might look real to non-domain practitioners, containing similar peaks and valleys to the original data, it must also contain information about real fuel material properties. For such a data generation process to be trusted further, these artificial representations of possible real samples must be evaluated by an expert. By training conditional GAN on information about Chemical Functional Groups (CFGs) present in existing fuel samples, we design a sample generation approach that generates spectroscopy of theoretical fuel with the desired chemical properties. We focus on synthesizing pure alkane mixtures that contain only CH<sub>2</sub> and CH<sub>3</sub> functional groups, as they are prominently distinguishable by the domain expert from other mixtures contained in the training set. We overlay these artificially generated samples with real pure alkanes from our dataset to visualize the difference. Then, we let the domain experts confirm their authenticity via the presence (or absence) of certain peaks in the expected regions of spectroscopy and their intensities. If successful, such a GAN sample generation core can be trusted further to produce new samples to enrich the existing dataset.

### 3.2.4 | Data

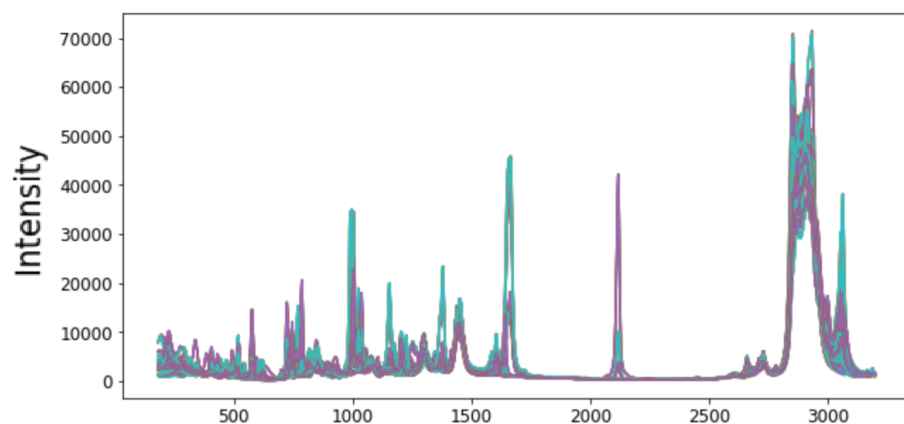
The spectroscopic data consists of two datasets, namely Raman and ATR, that were collected at the UIC High-Pressure Shock Tube and Lynch Laboratories using Raman and diamond Attenuated Total Reflectance (ATR) spectrometers accordingly. The Raman dataset consists of 245 observations from 49 unique fuel samples, as discussed in Chapter §2.3.2, while the ATR data has 145 observations from 29 fuel mixtures. Each ATR sample has close to 60,000 features, which are absorbance/intensity values recorded across the entire wavelength range between  $[400, 4000] \text{ cm}^{-1}$ . The data are processed to filter out noisy, uninformative regions between 1800 and 2500 wavenumbers. We reduce the number of features via cubic spline interpolation to 1562, as in Raman spectra, for consistency and without any major resolution quality loss. ATR data are then split into training, and testing sets 80/20 for ML efficacy testing. For domain-expert conformance checking, we also acquire information about the known CFG concentrations for ATR samples. These concentrations represent ratios from extracted UNIFAC functional groups for pure components and mixtures that were collected at the Lynch Laboratory. The resulting array consists of the fraction of concentrations (scalar values) that sum up to 1 for each mixture sample. In this study, we only use CH<sub>2</sub> and CH<sub>3</sub> functional group concentrations for the collected 145 ATR samples. Since pure alkanes only consist of CH<sub>2</sub> and CH<sub>3</sub> groups, the resulting concentration array consists of pairs of values (i.e.,  $[0.4, 0.6]$ ). This information is later used for the final stage of model evaluation.

## 3.3 | Experimental Results

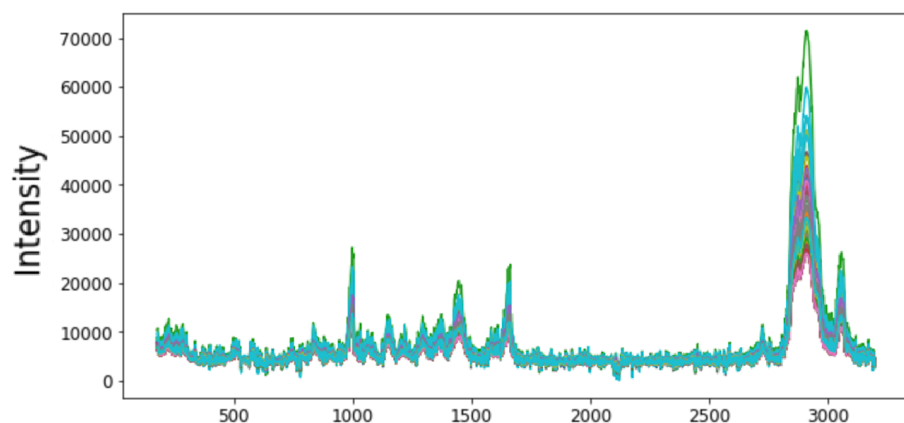
The initial GAN model is first trained on Raman spectra data using classic “Vanilla” architecture with appropriate network capacity capable of holding the com-

pressed knowledge from given spectroscopic data. We use two fully connected layers with 256 nodes for discriminator and generator. The choice of number of hidden layers and nodes is dictated by general guidelines throughout the GAN literature and following implementation by (122).

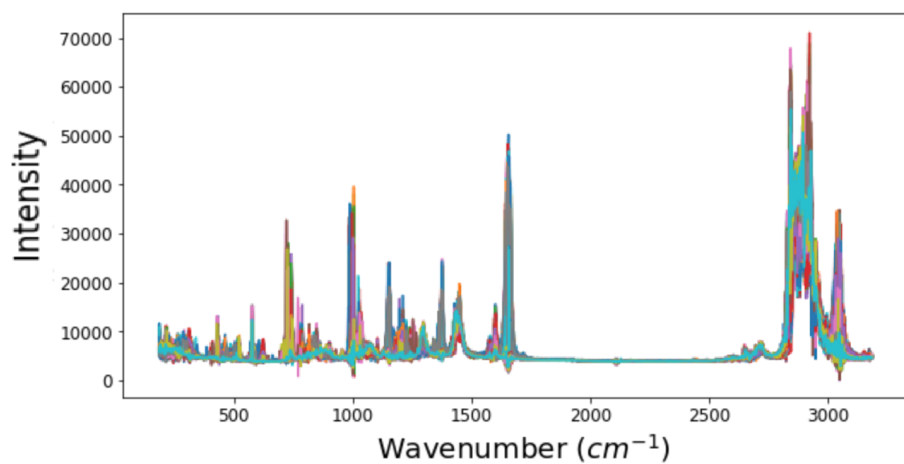
We use the ReLU activation function for all layers and binary cross-entropy loss function to calculate the gradient for differentiating between real and synthesized samples incorrectly. We also employ minibatch discrimination and examine how changing latent space parameters affects the model performance. Finally, we artificially add statistical noise to the system to improve training stability using smooth labeling. Instead of using hard binary labels for identifying real and synthetic samples, we apply a small random offset to the label and penalize the discriminator when the prediction for any real samples exceeding a certain probability threshold (i.e., 0.9 instead of 1), thereby promoting sparse activations. The resulting  $S_{synth}$  data scaling and overall resemblance to  $S_{real}$  remains low (Figure 3.2 (a, b)). The model is highly unstable and sensitive to gradient updates, partially because data are not normalized and hard to process for neural network activation functions in the original range of values. Increasing latent data dimension size from 128 dimensions to the size of training data, which is 1562 dimensions, slightly improves the quality of generated samples. However, the output range for absorbance values is still 1,000 orders of magnitude smaller than for real data. Therefore, we employ mode-specific data normalization to address the scaling and immediately observe improved model convergence towards realistic data scaling and proportions, as can be seen in Figure 3.2(c).



(a) Real



(b) Vanilla GAN



(c) MS-norm GAN

Figure 3.2: Comparison between (a) real Raman training data and synthetic data generated using trained (b) Vanilla GAN model (c) GAN with Mode Specific (MS) normalization

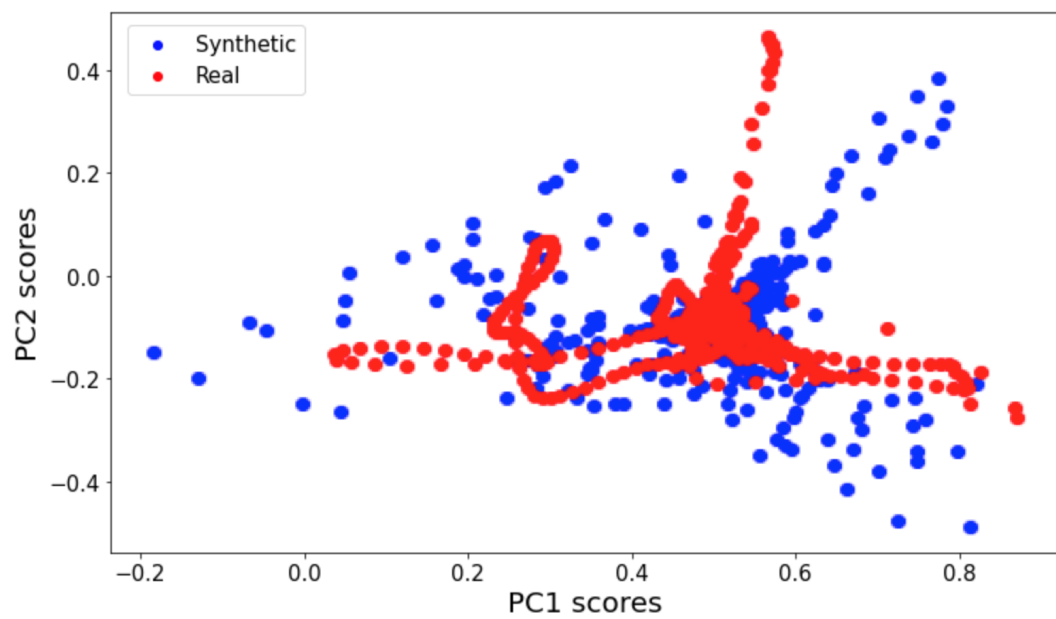


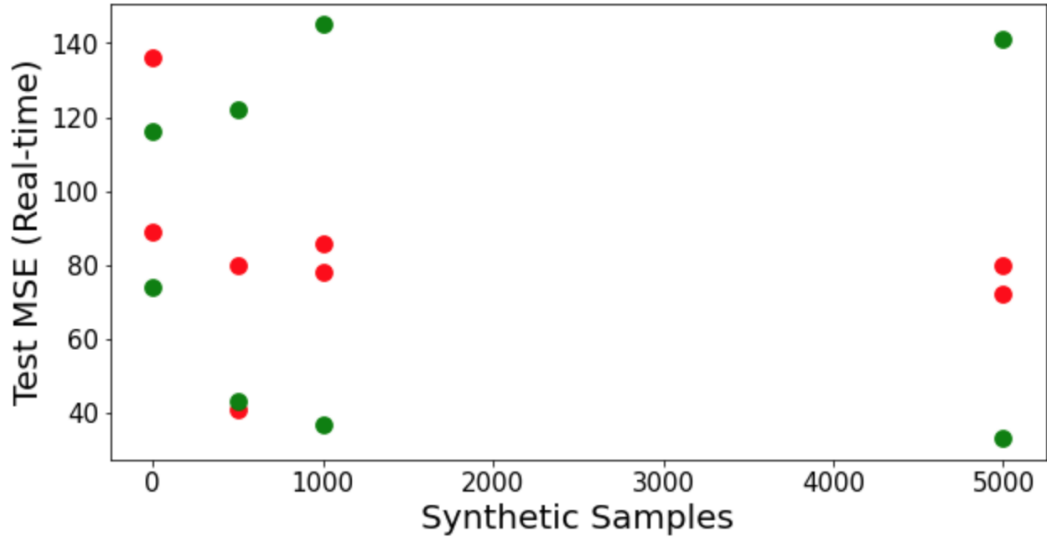
Figure 3.3: PCA first and second principal component overlay between real training spectra and synthetic spectra generated via GAN with mode specific normalization

		Component					
	USE	Conditional Label	Discriminatory steps	Gumbel-Softmax	Packing	BCE loss	Gradient Penalty
Avg	MSE (Test)	220.58	229.19	224.21	203.07	297.22	221.20
	KS Score	0.316	0.356	0.353	0.325	0.286	0.366
	CN Label Min	36.9	36.1	34.6	38.7	25.1	34.4
	CN Label Max	48.1	47.4	50.5	50.5	37.3	45
	Diff CN	11.2	11.2	15.9	11.8	12.1	10.5
	NOT USE						
	MSE (Test)	241.52	232.91	237.89	259.03	164.88	240.9
	KS Score	0.32	0.279	0.282	0.311	0.349	0.269
	CN Label Min	36.7	37.4	40.7	34.9	48.8	39.2
	CN Label Max	48.2	48.8	45.8	45.8	59.4	51.4
	Diff CN	11.5	11.4	6.9	10.8	10.5	12.1

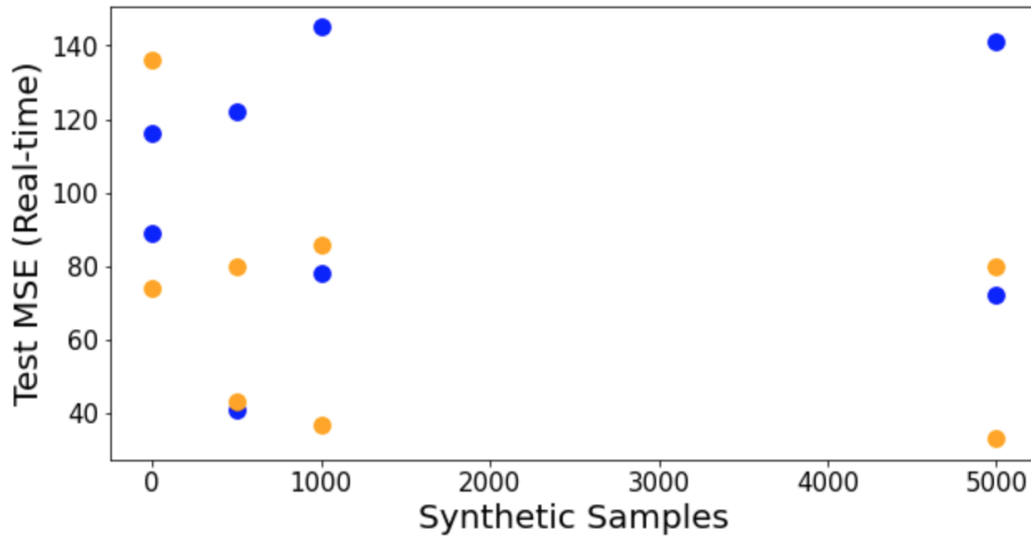
Table 3.1: ML Ablation test for Advanced GAN components



To evaluate the quality of the generated data, we first consider the resulting  $S_{synth}$  KS score. The newly generated data has  $KS_{score} = 0.536$  indicating that artificial and real data CFDs are partially similar. We also use PCA decomposition to project the first two PCs of each data subset onto each other (Figure. 3.3) to observe if two subsets are decomposed similarly. The explained variance (EV) by the first component  $PC_1$  is 43.77 and 38.27% for synthetic and real data, correspondingly. The EV for  $PC_2$  is recorded as 27.01 and 33.48%. The two subsets can be concluded to be similar, as their projections to lower-dimensional manifold carry almost identical information. Next, we evaluate the performance of our fitted GAN using model efficacy test and observe yet high  $MSE_{synth} = 119$  explained in the evaluation section compared to the  $MSE_{base} = 14.8$ , which indicates further improvement is necessary. To understand the effect of statistical noise associated with the choice of the prediction model and labeling oracle for model efficacy evaluation, we compare the use of the two most advanced model architectures from §2.3.3. Using both the Fine-tuned Neural Network and the Fine-tuned Support Vector Regression models as labeling oracle and final CN prediction model, we record their corresponding testing MSEs to establish optimal model combination. We establish that using the Fine-tuned SVR model as both labeling oracle (Figure 3.4(a)) and final prediction model (Figure 3.4(b)) leads to lowest testing error, and is therefore an optimal model for both use cases. We perform a comprehensive ablation test to further explore the value of using various advanced GAN techniques described in §3.2.2. We conduct the test by iterative removal of components to understand their contribution to overall model performance. We consider various metrics, such as model efficacy, generated label range, diversity, and standard statistical tests, to evaluate the change in GAN behavior with and without these components (Table 3.1). Due to the high stochastic nature and computational power required to train GAN, a



(a) Oracle evaluation. SVR(green), NN(red)



(b) Prediction model evaluation. SVR(orange), NN(blue)

Figure 3.4: CN Prediction and synthetic data Labeling accuracy measured for both SVR and NN models. Four combinations of Oracle-Prediction models tested on Real-time data, (a) Oracle and (b) Prediction model evaluation

comprehensive ablation test for all possible combinations of components is not feasible within a given amount of time. Therefore we rely on general patterns observed during the limited series of tests (total of 114 model configurations trained), to fix multiple parameters and advance the architecture complexity incrementally afterwards.

The complete set of modules and parameter considerations for ablation testing employs the following modifications. We use mode-specific normalization, which includes conditioning both generator and discriminator on a discrete GMM modes and activate them separately from continuous spectra attributes using Gumbel-Softmax activation. In addition, the transformed absorbance/intensity values disseminate through a chain of Leaky ReLU activation functions in the hidden layers. The generator architecture is updated with Batch Normalization layers, and discriminator/critic is equipped with Dropout layers respectively after each activation step. The binary cross-entropy loss function is compared against the Earth-Movers distance transportation function, effectively replacing discriminator with WGAN critic configuration. Gradient penalty is used instead of weight clipping for improved training stability. Additionally, we evaluate packing for the discriminator model, multi-step critic model update (as discussed in §3.2.2), and several variants of conditioning GAN on additional information, which we shall discuss later.

Building on top of our initial observations and ablation test results (Table 3.1) we observe the following. After applying mode-specific normalization, the transformed spectroscopic data now holds information on each wavenumber's underlying Gaussian distribution mode and has an optimal value range for efficient deep network modeling. We also confirm that using binary cross-entropy loss function for discriminator leads to a mode collapse, as the output of generator becomes uniform and lacks diversity in output samples. Hence, it results in

higher model efficacy MSE and a narrow range of CN values after labeling the samples. Therefore, the optimal GAN model for synthesizing both Raman and ATR spectroscopy is found to consist of generator and critic models, known as WGAN, with several other modifications discussed further. Using conditional information, such as CN labels as part of the ablation test, also shows improvement in the generated data diversity. Although packing multiple samples into one observation using the PacGAN approach improves the model convergence for the baseline discriminator architecture, it is not practical to use packing for Conditional GAN configuration with W-loss function. While it is straightforward to pack samples for the standard discriminators based on whether they are real or synthetic samples, labeled 1 or 0, using the similarity loss function negates this option. To maintain mixture-specific information, such packing procedures must be selected, ensuring that the same mixtures are packed together. However, due to the disparity in sample sizes across mixtures, determining the ideal packing degree is a difficult task. Further, having an imbalanced number of mixtures in each pack can cause fluctuations in the learning rate of the generator and critic, skewing learning towards particular mixtures and causing overall training instability. Therefore, packing is not used to construct the optimal model. Changes to the latent space initialization method, size, or network relative processing capability are not required at this stage because they do not induce any changes in results and are thus not recorded or evaluated further. We also note that the oscillations of the training loss are stabilized when a 2-4 step critic update is performed before calculating and propagating gradients back to the network weights of the generator model. Based on the review of ablation test results, we finally conclude that the optimal spectra GAN model has Conditional WGAN architecture with two layers of Leaky ReLU activations, Batch Normalization, and Dropout hidden layers. The optimal model utilizes mode-

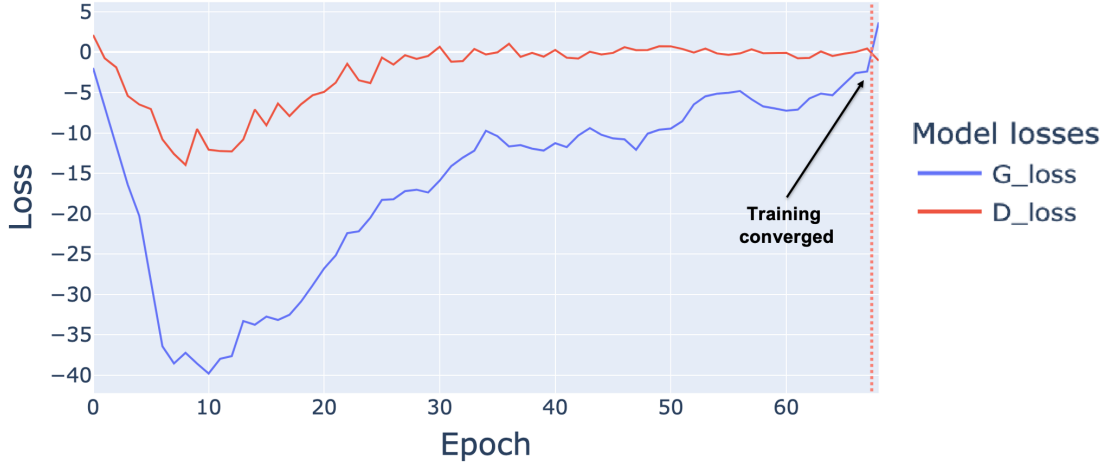


Figure 3.5: Optimal GAN training loss history.

specific normalization with Gumbel-Softmax and hyperbolic tangent (htan) as the final output layer activations for discrete and continuous values, respectively. Moreover, our optimal GAN model uses multi-step critic update and gradient penalty instead of weight clipping for efficient gradient update.

To measure the effectiveness of the optimal GAN model for spectroscopic data, we first evaluate the training process convergence by tracking generator and critic losses. We observe that after 66 epochs, the generator and critic loss functions converge as shown in Figure 3.5, indicating that we achieve a stable trained model. Next, we apply statistical similarity and ML efficacy tests to estimate the quality of the newly generated data. Recording distribution similarity between  $S_{real}$  and  $S_{synth}$  generated data results in a  $KS_{score} = 0.748$ , which is a significant improvement. Plotted over the entire spectrum, Figure 3.8, we observe that lower KS-score locations correspond to locations of the highest spectral peaks, hence, the highest variance in data, as expected. The model efficacy evaluation results in  $MSE_{synth} = 37.9$  with a standard deviation of  $\sigma = 4.1$ , indicating a significant improvement over the initial GAN configuration that resulted in  $MSE_{synth} = 119$ . Also, adopting the Mixed data scenario (described in

§2.3.2), training GAN on all 245 available Raman samples did not improve prediction accuracy over baseline. The range of CN values generated for synthetic data as part of the Weak labeling process ( $[19.5 - 80.5]$ ) revealed even more variability in the data produced and the overall quality of the generating process.

To evaluate the conformance of our sample generation core with the domain knowledge, we enforce the generator to produce samples of a certain alkane mixture. In particular, we condition our GAN on given CH<sub>2</sub> and CH<sub>3</sub> chemical functional group concentrations encountered in such alkane. The concentration values provided by Lynch Laboratory were only available for the ATR dataset. Therefore we train GAN on ATR data instead of Raman for this evaluation. The network architecture is updated to introduce an additional dimension that holds spectroscopic data and two concentration values as three unique channels (dimensions) of information (Figure 3.7, 3.6). This allows GAN to concurrently process and condition the generator to produce spectroscopy based on the provided concentration labels. Consequently, the critic also analyzes input samples regarding their corresponding labels to calculate the difference in similarity between artificial and real samples.

Both the generator and critic models, learn the underlying data distribution and are able to construct compact representations of real spectroscopy to adjust the generation of new samples according to input concentration values. Ultimately, we train our GAN on 145 available ATR samples and enforce the trained GAN generator to generate pure alkane samples by providing conditional concentrations of CH<sub>2</sub> and CH<sub>3</sub> as a tuple of proportion float values (i.e.,  $[0.35, 0.65]$ ). Note that pure alkanes are defined as mixtures, the sum of which CH<sub>2</sub> and CH<sub>3</sub> concentrations add up to 1. The synthetic pure alkane samples (Figure 3.9) are then visually evaluated by overlaying real  $S_{real}$  and generated  $S_{synth}$ , and inspecting the important spectroscopy “peaks” learned by GAN. The

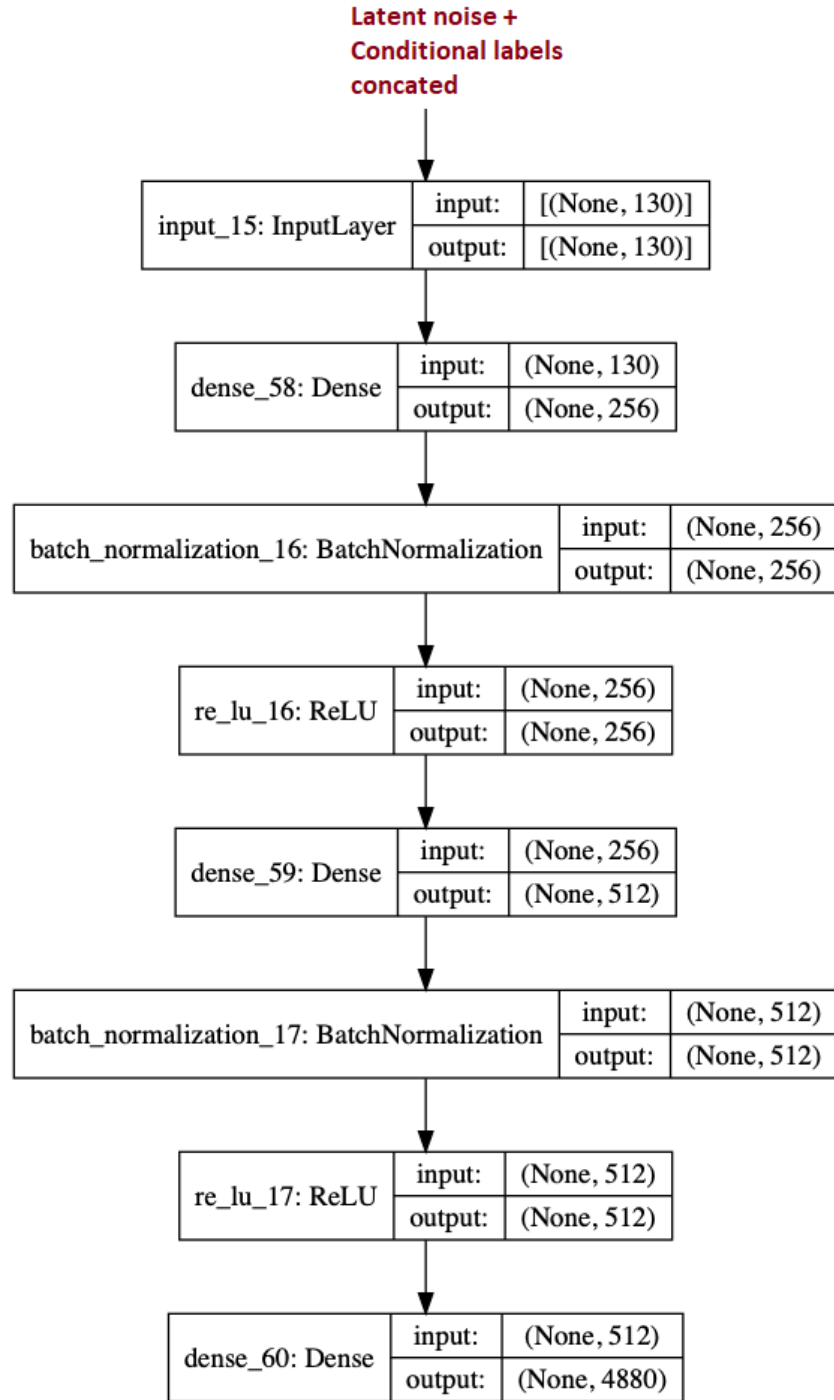


Figure 3.6: GAN Spectra generator optimized architecture.

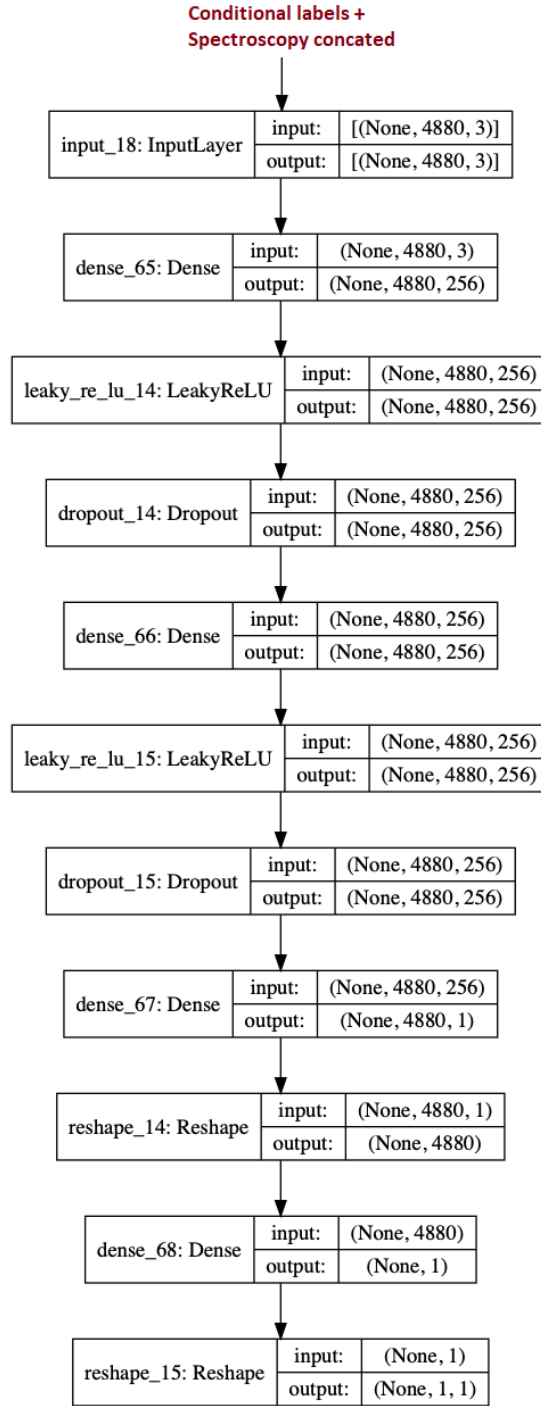


Figure 3.7: CGAN Spectra critic optimized architecture.



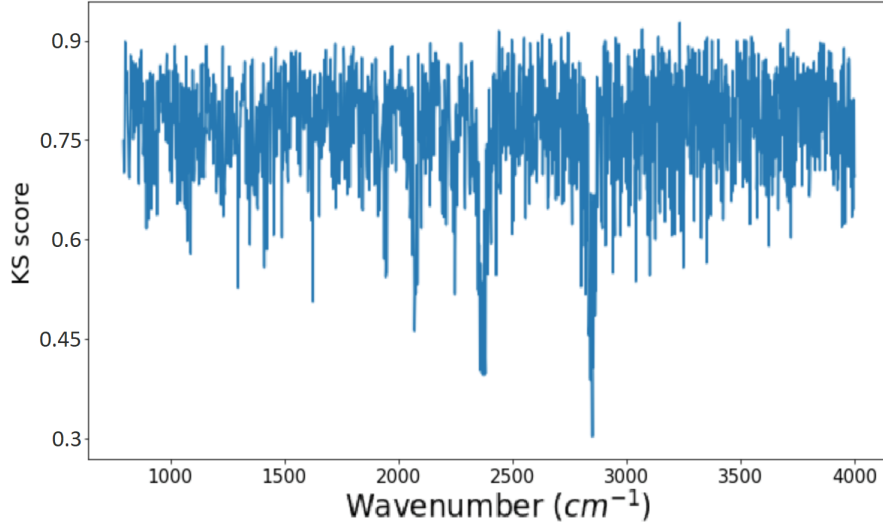


Figure 3.8: The KS score for each wavenumber for CGAN output.

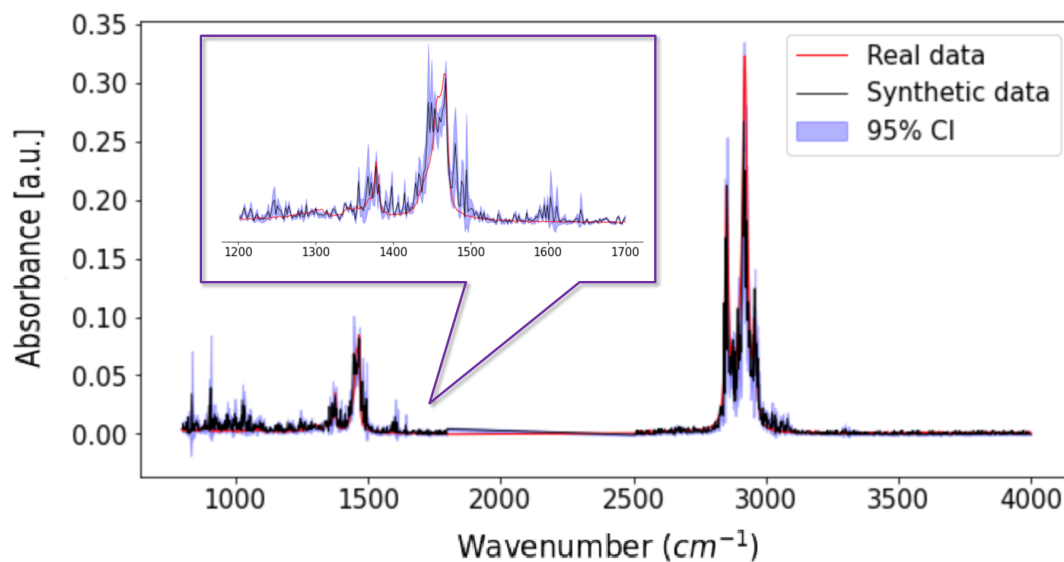
synthetic data are considered acceptable, as it captures important regions associated with the underlying properties of such a mixture with a low variance level (the purple band shows the 95% Confidence Interval). It should, however, be noted that due to the presence of other mixtures in the data we may observe noisy patterns. In addition, since the number of observations is limited particularly for a certain “classes” of mixtures, disproportional learning of certain mixtures over others is resulted. Furthermore, after performing model efficacy test on the GAN conditioned on the concentration information, we observe improvement,  $MSE_{synth} = 14.04$ , over ATR with  $MSE_{base} = 25.76$ . This leads to a reasonable assumption that by expanding the diversity of the existing dataset with synthetic samples produced by a robust generator, we are able to improve CN prediction accuracy and improve the generalization capacity on unforeseen samples. Since using GAN without CFG concentration labels did not result in improvement over  $MSE_{base}$  for the Raman dataset, we also conclude that domain-driven conditional information is a necessary tool to guide GAN towards meaningful differentiation of samples. GAN can operate and diversify samples in an extra

dimension of representation by providing additional details regarding empirical properties of data.

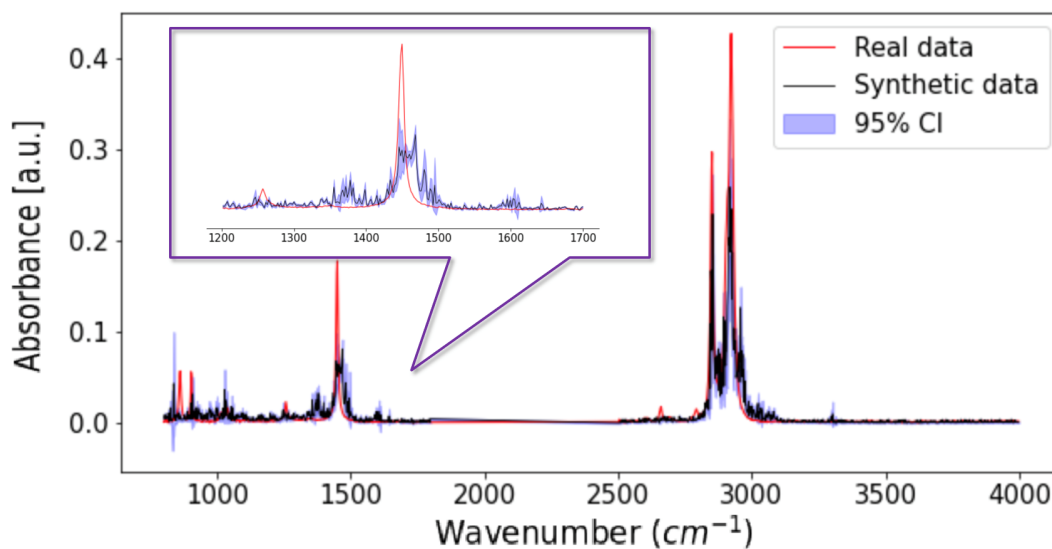
Moving one step further, we restructure our generator as a multi-output model that, besides generating new spectroscopic samples, outputs the estimated remaining concentration of unknown chemical functional groups. This serves as a sanity check of whether our GAN does indeed condition spectroscopy on “known” functional group concentrations. Since we know that for all mixtures the remaining concentration levels should equal to  $1 - CH_2 - CH_3$ , which is 0 for all pure alkanes, we compare the predicted remaining concentrations of generated samples to their theoretical remainder values (Table 3.2). We produce 10,000 random samples using the trained conditioned multi-output GAN, record the predicted output and averaging the difference between predicted and expected remaining concentration values across all generated samples. As a result, we estimate a mean prediction error of 0.2572. In other words, from a Domain-expert standpoint, the prediction accuracy of the sum of remaining CFG concentrations for 10,000 generated samples is 74.28%. This result further supports the claim that conditioning GAN on real fuel chemical properties allows GAN to successfully map those properties to spectroscopy and generate realistic artificial fuel samples.

<b>Fuel</b>	<b>CH2</b>	<b>CH3</b>	<b>1-CH2-CH3</b>	<b>Predicted Remaining</b>	<b>Error</b>
Alk 1	0.848	0.152	0.000	0.251	0.251
Alk 2	1.000	0.000	0.000	0.154	0.154
Alk 3	0.700	0.300	0.000	0.038	0.038
Alk 4	0.867	0.133	0.000	0.172	0.172
Alk 5	0.823	0.177	0.000	0.071	0.071
Alk 6	0.808	0.192	0.000	0.106	0.106
Alk 7	0.789	0.211	0.000	0.140	0.140
Mix 1	0.293	0.315	0.392	0.313	0.079
Mix 2	0.678	0.226	0.096	0.019	0.077
Mix 3	0.767	0.168	0.065	0.150	0.085

Table 3.2: Remaining concentration prediction result for first 10 mixtures

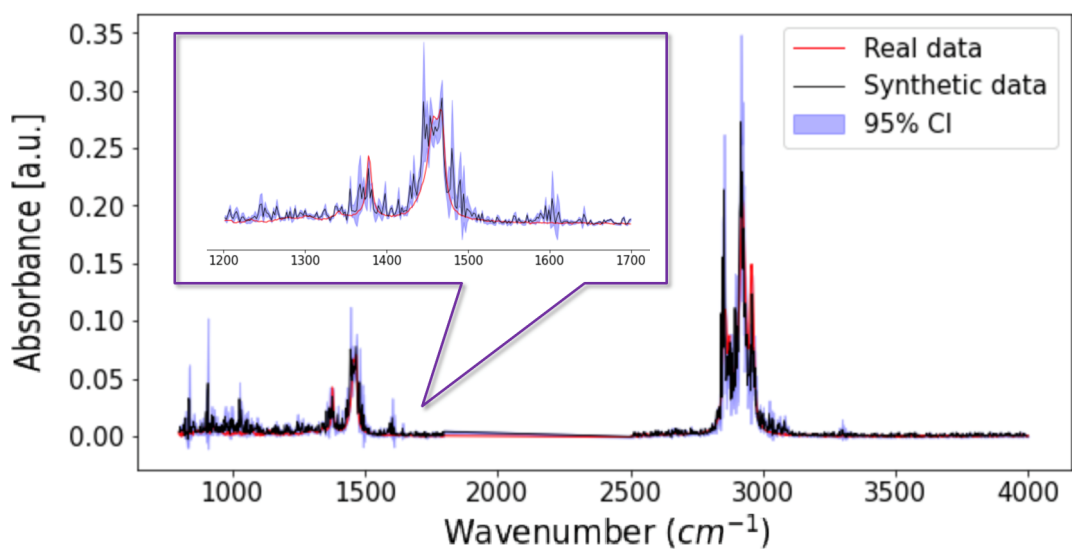


(a) Alk 1

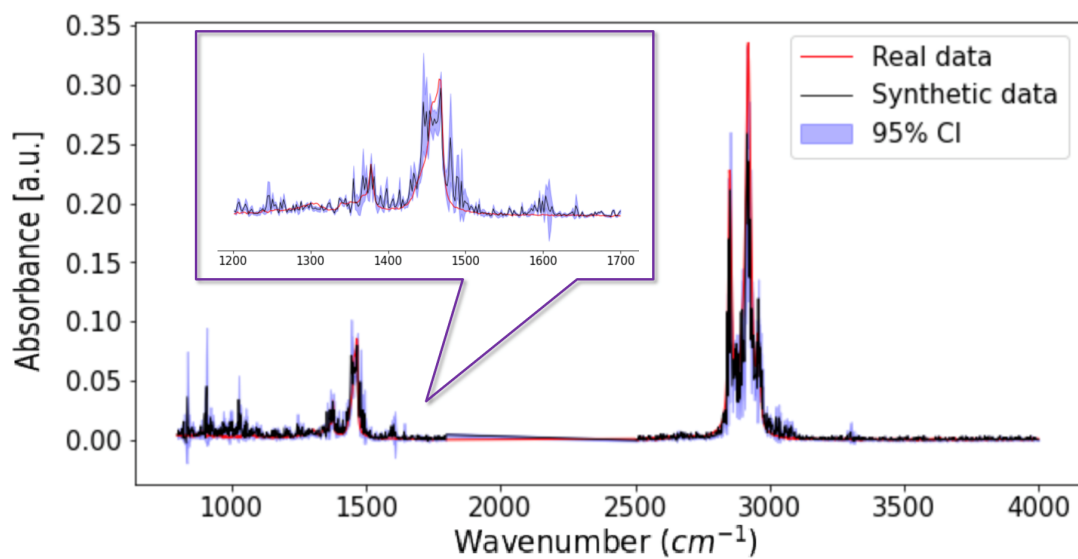


(b) Alk 2

Figure 3.9: Pure alkane Mixtures ATR comparison between Real (red) and generated (black) data with 95% Confidence Interval (blue) across generating 10,000 samples of given spectra. CH<sub>2</sub>-CH<sub>3</sub> concentration (a) Alk 1: [0.848, 0.152] (b) Alk 2: [0,1]

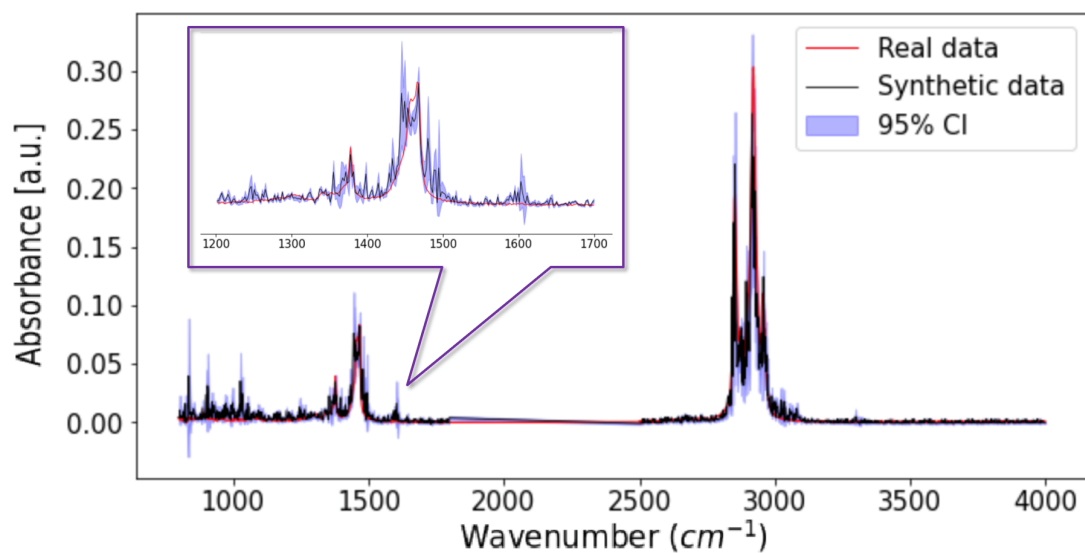


(c) Alk 3

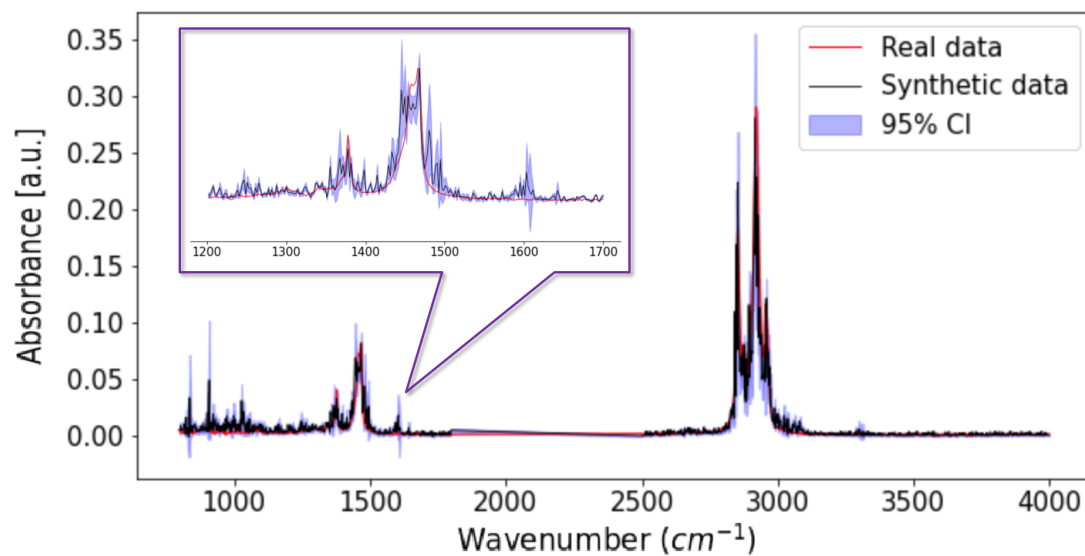


(d) Alk 4

Figure 3.9: Pure alkane Mixtures ATR comparison between Real (red) and generated (black) data with 95% Confidence Interval (blue) across generating 10,000 samples of given spectra. CH<sub>2</sub>-CH<sub>3</sub> concentration (c) Alk 3: [0.7, 0.3] (d) Alk 4: [0.867, 0.133]

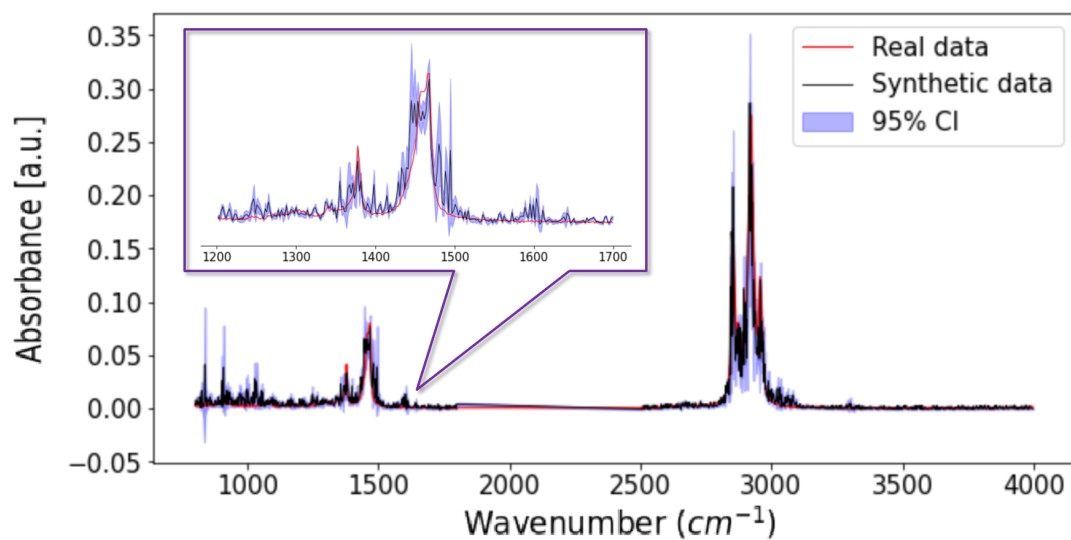


(e) Alk 5



(f) Alk 6

Figure 3.9: Pure alkane Mixtures ATR comparison between Real (red) and generated (black) data with 95% Confidence Interval (blue) across generating 10,000 samples of given spectra. CH<sub>2</sub>-CH<sub>3</sub> concentration (e) Alk 5: [0.823, 0.177] (f) Alk 6: [0.808, 0.192]



(g) Alk 7

Figure 3.9: Pure alkane Mixtures ATR comparison between Real (red) and generated (black) data with 95% Confidence Interval (blue) across generating 10,000 samples of given spectra. CH<sub>2</sub>-CH<sub>3</sub> concentration (g) Alk 7: [0.789, 0.211]

## Conclusion

This research has been initiated with the assumption that there is indeed a deep connection between fuel mixture ignition properties and the corresponding fuel spectroscopy pattern. We reasonably assumed that such a connection could be compressed and learned using Machine Learning tools. To test this assumption, we developed a comprehensive methodology to evaluate such mapping tools grounded in explainability and domain-expert evaluation for complete result transparency. We tasked ourselves with solving a real-world chemometrics and engineering problem and provided a detailed description of the theoretical background and implementation considerations for all techniques used. In the process of exploring this relation, we have also identified a gap in the literature on scalability and interpretability approaches for ML-based analysis and synthetic data generation of spectroscopy data. Given the high-dimensionality and limited size nature of spectroscopy data, we explored the option of generating additional synthetic samples to expand our training dataset size and diversity. With the lack of quality research on the technical implementation of generative modeling for spectroscopy data, we implemented and evaluated the best available techniques to build a robust, transparent synthetic spectroscopy simulation core.



Using various feature selection techniques, we were able to confirm that the reduced subset of features in general results in a less complex, computationally more efficient, and yet accurate prediction model. Based on our interpretability evaluation, we were also able to confirm the conformance of the attribute selection and prediction results in terms of known chemistry, ensuring that the domain expert can trust such algorithmic results. We were able to explain the behavior of complex ML models using various model-agnostic methods, which has not been done before in chemometrics to this extent, and report on the performance of model-based methods. As discovered, using model-agnostic explanation techniques, features derived from complex ML models were generally of higher quality than from model-based methods, with the exception of a subset of Random Forest features that performed best overall on the Performance-Explainability trade-off scale. While the PCA and PLS decomposition methods are still considered primary and most widely-used techniques for spectroscopy data analysis, our investigation concluded that these methods were inferior to all other feature extraction techniques covered in this work to establish an accurate relationship between fuel spectroscopy and associated CN. This claim does not necessarily extend to all possible processes considered in chemometrics, as PCA and PLS have been shown to function effectively for other applications.

In the second part of this research, we developed a robust synthetic sample generation model to address the issue of the limited size of fuel spectroscopy data. By analyzing and testing the most advanced generative techniques, we were able to construct a powerful GAN model capable of synthesizing high-quality, artificial Raman and ATR fuel spectroscopy data. Such a model was shown to not only improve data representation but also boost the performance of our optimal CN prediction model, resulting in almost twice better prediction accuracy. Through a series of comprehensive statistical similarity, ML efficacy,

and domain-expert conformance evaluations, we were also able to show that such data are generated in a meaningful, explainable way and hold the same physical properties as real data.

All in all, from isolating particular regions in spectroscopy using statistical explanation behind the selection process to mapping underlying spectroscopy distribution with GAN, we were able to develop an explainable framework of spectroscopy to fuel property mapping. Our results demonstrate that such a connection exists and can be efficiently compressed and represented to predict empirical sample properties. As every step of this evaluation remains explainable, such methods are likely to be trusted and used by domain practitioners, which we encourage.

Based on the provided discussion, a further investigation into the efficacy of employing the conditional GAN model to generate Raman spectroscopy is necessary. While we show that conditioning GAN on CFG concentrations improves prediction accuracy for ATR data, the same claim cannot be fully concluded for Raman data since no CFG concentrations were not available for mixtures in the Raman dataset. Future work must also include a comprehensive analysis of synthesizing reduced data based on the regions selected with feature selection methods, described in Chapter §2, as well as the generated reduced data for CN prediction. An analysis of signal filtering and spectroscopy pre-processing methods must also accompany such investigation to ensure a complete end-to-end solution from spectroscopy collection to CN prediction is developed.

In practice, such framework can be deployed as a complete AI solution for mapping material properties to material spectroscopy and generating novel materials (products) with desired properties that have not been designed or explored yet. The proposed solution and the accompanying analysis, plus methodology, are not limited to merely fuel spectroscopy domain either. The provided

insight and findings of this study can be generalized to other domains as well. Using automated explainability metrics and effective data generation process a pharmaceutical, food or energy industry expert can easily decide on the quality of newly generated design or property prediction. The decision to refuse or accept new outcome becomes less arbitrary. Instead, it is based on data analysis. Since such framework can be employed for any type of spectroscopy and underlying material in question, it can be scaled across entire organisation or span across multiple industries, saving time and money to evaluate and develop new and existing products.

# Elsevier License Agreement

## *License of publishing rights*

I hereby grant to Journal Owner an irrevocable non-exclusive license to publish, distribute and otherwise use all or any part of the manuscript identified above and any tables, illustrations or other material submitted for publication as part of the manuscript (the “Article”) in all forms and media (whether now known or later developed), in all languages, throughout the world, for the full term of copyright, and the right to license others to do the same, effective when the Article is accepted for publication. I acknowledge the importance of the integrity, authenticity and permanence of the scholarly record and agree that the version of the Article that appears or will appear in the journal and embodies all value-adding publisher activities (including peer review co-ordination, copy-editing, formatting, (if relevant) pagination, and online enrichment) shall be the definitive final record of published research (“the Published Journal Article”). I further acknowledge and agree that nothing in this Agreement shall be deemed to permit redundant/duplicate publication of the Article in violation of publishing ethics principles, as further described below.

## *Supplemental Materials*

“Supplemental Materials” shall mean materials published as a supplemental

part of the Article, including but not limited to graphical, illustrative, video and audio material. With respect to any Supplemental Materials that I submit, Journal Owner shall have a perpetual worldwide, non-exclusive right and license to publish, extract, reformat, adapt, build upon, index, redistribute, link to and otherwise use all or any part of the Supplemental Materials, in all forms and media (whether now known or later developed) and permit others to do so. The publisher shall apply the same end user license to the Supplemental Materials as to the Article where it publishes the Supplemental Materials with the Article in the journal on its online platforms on an Open Access basis.

*Research Data*

“Research Data” shall mean the result of observations or experimentation that validate research findings and that are published separate to the Article, which can include but are not limited to raw data, processed data, software, algorithms, protocols and methods. With respect to any Research Data that I wish to make accessible on a site or through a service of Journal Owner, Journal Owner shall have a perpetual worldwide, non-exclusive right and license to publish, extract, reformat, adapt, build upon, index, redistribute, link to and otherwise use all or any part of the Research Data in all forms and media (whether now known or later developed), and permit others to do so. Where I have selected a specific end user license under which the Research Data is to be made available on a site or through a service, the publisher shall apply that end user license to the Research Data on that site or service. Scholarly communication rights. I understand that I retain the copyright in the Article and that no rights in patents, trademarks or other intellectual property rights are transferred to the Journal Owner. As the author of the Article, I understand that I shall have the same rights to reuse the Article as those allowed to third party users (and Journal Owner) of the Article under the CC BY License. User rights. The publisher

will apply the Creative Commons Attribution 4.0 International License (CC BY) to the Article where it publishes the Article in the journal on its online platforms on an Open Access basis. For further information, see

<http://www.elsevier.com/about/open-access/open-access-options>.

The CC BY license allows users to copy, to create extracts, abstracts and new works from the Article, to alter and revise the Article and to make commercial use of the Article (including reuse and/or resale of the Article by commercial entities), provided the user gives appropriate credit (with a link to the formal publication through the relevant DOI), provides a link to the license, indicates if changes were made and the licensor is not represented as endorsing the use made of the work. The full details of the license are available at

<https://creativecommons.org/licenses/by/4.0/> Reversion of rights. Articles may sometimes be accepted for publication but later rejected in the publication process, even in some cases after public posting in “Articles in Press” form, in which case all rights will revert to the author (see

<http://www.elsevier.com/locate/withdrawalpolicy>). Revisions and addenda.

I understand that no revisions, additional terms or addenda to this License Agreement can be accepted without Journal Owner’s express written consent.

I understand that this License Agreement supersedes any previous agreements

I have entered into with Journal Owner in relation to the Article from the date

hereof. Copyright Notice. The publisher shall publish and distribute the Ar-

ticle with the appropriate copyright notice. I agree that any copy of the Article or any part of the Article that I distribute or post will include such copy-

right notice and a full citation to the Published Journal Article. Author Representations/Ethics and Disclosure. I affirm the Author Representations noted

below, and confirm that I have reviewed and complied with the relevant Instructions to Authors, Ethics in Publishing policy, Declarations of Interest dis-

closure, and information for authors from countries affected by sanctions (Iran, Cuba, or Syria ). Author Representations. The Article I have submitted to the journal for review is original, has been written by the stated authors and has not been previously published. The Article was not submitted for review to another journal while under review by this journal and will not be submitted to any other journal. The Article and the Supplemental Materials do not infringe any copyright, violate any other intellectual property, privacy or other rights of any person or entity, or contain any libellous or other unlawful matter. I have obtained written permission from copyright owners for any excerpts from copyrighted works that are included and have credited the sources in the Article or the Supplemental Materials. Except as expressly set out in this License Agreement, the Article is not subject to any prior rights or licenses. If I and/or any of my co-authors reside in Iran, Cuba, or Syria, the Article has been prepared in a personal, academic or research capacity and not as an official representative or otherwise on behalf of the relevant government or institution. If I am using any personal details or images of patients, research subjects or other individuals, I have obtained all consents required by applicable law and complied with the publisher's policies relating to the use of such images or personal information. See <http://www.elsevier.com/patientphotographs> for further information. Any software contained in the Supplemental Materials is free from viruses, contaminants or worms. If the Article or any of the Supplemental Materials were prepared jointly with other authors, I have informed the co-author(s) of the terms of this License Agreement and that I am signing on their behalf as their agent, and I am authorized to do so Governing Law and Jurisdiction. This License Agreement will be governed by and construed in accordance with the laws of the country or state of Journal Owner ("the Governing State"), without regard to conflict of law principles, and the parties ir-

revocably consent to the exclusive jurisdiction of the courts of the Governing State. For information on the publisher's copyright and access policies, please see <http://www.elsevier.com/copyright>.



---

## References

- [1] F. Akulich, H. Anahideh, M. Sheyyab, and D. Ambre, “Explainable predictive modeling for limited spectral data,” *Chemometrics and Intelligent Laboratory Systems*, p. 104572, 2022.
- [2] K. Pearson, “Mathematical contributions to the theory of evolution, on the law of ancestral heredity,” *Proceedings of the Royal Society of London*, vol. 62, no. 379-387, pp. 386–412, 1898.
- [3] N. S. Altman, “An introduction to kernel and nearest-neighbor nonparametric regression,” *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.
- [4] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, *et al.*, “Searching for mobilenetv3,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1314–1324, 2019.
- [5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.
- [6] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, “Mastering the game of go with deep neural networks and tree search,” *nature*, vol. 529, no. 7587, pp. 484–489, 2016.
- [7] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, *et al.*, “Mastering chess and shogi by self-play with a general reinforcement learning algorithm,” *arXiv preprint arXiv:1712.01815*, 2017.

- [8] A. W. Senior, R. Evans, J. Jumper, J. Kirkpatrick, L. Sifre, T. Green, C. Qin, A. Žídek, A. W. Nelson, A. Bridgland, *et al.*, "Improved protein structure prediction using potentials from deep learning," *Nature*, vol. 577, no. 7792, pp. 706–710, 2020.
- [9] A. G. Abdul Jameel, N. Naser, A.-H. Emwas, S. Dooley, and S. M. Sarathy, "Predicting fuel ignition quality using 1h nmr spectroscopy and multiple linear regression," *Energy & Fuels*, vol. 30, no. 11, pp. 9819–9835, 2016.
- [10] A. G. A. Jameel, N. Naser, G. Issayev, J. Touitou, M. K. Ghosh, A.-H. Emwas, A. Farooq, S. Dooley, and S. M. Sarathy, "A minimalist functional group (mfg) approach for surrogate fuel formulation," *Combustion and Flame*, vol. 192, pp. 250–271, 2018.
- [11] M. Dahmen and W. Marquardt, "A novel group contribution method for the prediction of the derived cetane number of oxygenated hydrocarbons," *Energy & Fuels*, vol. 29, no. 9, pp. 5781–5801, 2015.
- [12] Y.-H. Kwon and M.-G. Park, "Predicting future frames using retrospective cycle gan," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1811–1820, 2019.
- [13] Z. Che, Y. Cheng, S. Zhai, Z. Sun, and Y. Liu, "Boosting deep learning risk prediction with generative adversarial networks for electronic health records," in *2017 IEEE International Conference on Data Mining (ICDM)*, pp. 787–792, IEEE, 2017.
- [14] M. Frid-Adar, I. Diamant, E. Klang, M. Amitai, J. Goldberger, and H. Greenspan, "Gan-based synthetic medical image augmentation for increased cnn performance in liver lesion classification," *Neurocomputing*, vol. 321, pp. 321–331, 2018.
- [15] K. Pearson, "Liii. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, vol. 2, no. 11, pp. 559–572, 1901.
- [16] H. Wold, "Estimation of principal components and related models by iterative least squares," *Multivariate analysis*, pp. 391–420, 1966.
- [17] L. Nørgaard, A. Saudland, J. Wagner, J. P. Nielsen, L. Munck, and S. B. Engelsen, "Interval partial least-squares regression (i pls): A comparative chemometric study with an example from near-infrared spectroscopy," *Applied Spectroscopy*, vol. 54, no. 3, pp. 413–419, 2000.

- [18] X. Zou, J. Zhao, and Y. Li, "Selection of the efficient wavelength regions in ft-nir spectroscopy for determination of ssc of 'fuji'apple based on bpls and fipls models," *Vibrational spectroscopy*, vol. 44, no. 2, pp. 220–227, 2007.
- [19] J.-H. Jiang, R. J. Berry, H. W. Siesler, and Y. Ozaki, "Wavelength interval selection in multicomponent spectral analysis by moving window partial least-squares regression with applications to mid-infrared and near-infrared spectroscopic data," *Analytical chemistry*, vol. 74, no. 14, pp. 3555–3565, 2002.
- [20] W. J. Murdoch, C. Singh, K. Kumbier, R. Abbasi-Asl, and B. Yu, "Interpretable machine learning: definitions, methods, and applications," *arXiv preprint arXiv:1901.04592*, 2019.
- [21] J. Berkson, "Application of the logistic function to bio-assay," *Journal of the American statistical association*, vol. 39, no. 227, pp. 357–365, 1944.
- [22] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and regression trees*. Routledge, 2017.
- [23] C. Molnar, *Interpretable machine learning*. Lulu. com, 2020.
- [24] G. Stiglic, P. Kocbek, N. Fijacko, M. Zitnik, K. Verbert, and L. Cilar, "Interpretability of machine learning-based prediction models in healthcare," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 10, no. 5, p. e1379, 2020.
- [25] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.
- [26] M. Du, N. Liu, and X. Hu, "Techniques for interpretable machine learning," *Communications of the ACM*, vol. 63, no. 1, pp. 68–77, 2019.
- [27] E. Al Ibrahim and A. Farooq, "Octane prediction from infrared spectroscopic data," *Energy & Fuels*, vol. 34, no. 1, pp. 817–826, 2019.
- [28] R. M. Balabin and S. V. Smirnov, "Variable selection in near-infrared spectroscopy: benchmarking of feature selection methods on biodiesel data," *Analytica chimica acta*, vol. 692, no. 1-2, pp. 63–72, 2011.

- [29] J. Andreu-Perez, L. L. Emberson, M. Kiani, M. L. Filippetti, H. Hagraš, and S. Rigato, "Explainable artificial intelligence based analysis for interpreting infant fnirs data in developmental cognitive neuroscience," *Communications biology*, vol. 4, no. 1, pp. 1–13, 2021.
- [30] M. T. Ribeiro, S. Singh, and C. Guestrin, "'why should i trust you?' explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144, 2016.
- [31] C.-Y. Wang, T.-S. Ko, and C.-C. Hsu, "Machine learning with explainable artificial intelligence vision for characterization of solution conductivity using optical emission spectroscopy of plasma in aqueous solution," *Plasma Processes and Polymers*, p. e2100096, 2021.
- [32] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 144–152, 1992.
- [33] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.
- [34] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.
- [35] T. K. Ho, "Random decision forests," in *Proceedings of 3rd international conference on document analysis and recognition*, vol. 1, pp. 278–282, IEEE, 1995.
- [36] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proceedings of the 31st international conference on neural information processing systems*, pp. 4768–4777, 2017.
- [37] E. Štrumbelj and I. Kononenko, "Explaining prediction models and individual predictions with feature contributions," *Knowledge and information systems*, vol. 41, no. 3, pp. 647–665, 2014.
- [38] S. O. Giwa, S. O. Adekomaya, K. O. Adama, and M. O. Mukaila, "Prediction of selected biodiesel fuel properties using artificial neural network," *Frontiers in Energy*, vol. 9, no. 4, pp. 433–445, 2015.

- [39] J. F. García-Martín, F. J. Alés-Álvarez, M. del Carmen López-Barrera, I. Martín-Domínguez, and P. Álvarez-Mateos, "Cetane number prediction of waste cooking oil-derived biodiesel prior to transesterification reaction using near infrared spectroscopy," *Fuel*, vol. 240, pp. 10–15, 2019.
- [40] M. Blanco, J. Coello, H. Iturriaga, S. MasPOCH, and C. De La Pezuela, "Near-infrared spectroscopy in the pharmaceutical industry," *Analyst London Royal Society of Chemistry*, vol. 123, pp. 135R–150R, 1998.
- [41] W. Plugge and C. Van Der Vliet, "The use of near infrared spectroscopy in the quality control laboratory of the pharmaceutical industry," *Journal of pharmaceutical and biomedical analysis*, vol. 10, no. 10-12, pp. 797–803, 1992.
- [42] Y. Roggo, P. Chaluz, L. Maurer, C. Lema-Martinez, A. Edmond, and N. Jent, "A review of near infrared spectroscopy and chemometrics in pharmaceutical technologies," *Journal of pharmaceutical and biomedical analysis*, vol. 44, no. 3, pp. 683–700, 2007.
- [43] E. Teye, X.-y. Huang, and N. Afoakwa, "Review on the potential use of near infrared spectroscopy (nirs) for the measurement of chemical residues in food," *Am. J. Food Sci. Technol*, vol. 1, pp. 1–8, 2013.
- [44] F. Van de Voort, "Fourier transform infrared spectroscopy applied to food analysis," *Food Research International*, vol. 25, no. 5, pp. 397–403, 1992.
- [45] H. Büning-Pfaue, "Analysis of water in food by near infrared spectroscopy," *Food Chemistry*, vol. 82, no. 1, pp. 107–115, 2003.
- [46] R. Jahani, H. Yazdanpanah, S. M. Van Ruth, F. Kobarfard, M. Alewijn, A. Mahboubi, M. Faizi, M. H. S. AliAbadi, and J. Salamzadeh, "Novel application of near-infrared spectroscopy and chemometrics approach for detection of lime juice adulteration," *Iranian Journal of Pharmaceutical Research: IJPR*, vol. 19, no. 2, p. 34, 2020.
- [47] P. Pandey, A. Rai, and M. Mitra, "Explainable 1-d convolutional neural network for damage detection using lamb wave," *Mechanical Systems and Signal Processing*, vol. 164, p. 108220, 2022.

- [48] S. Di Frischia, P. Giammatteo, F. Angelini, V. Spizzichino, E. De Santis, and L. Pomante, "Enhanced data augmentation using gans for raman spectra classification," in *2020 IEEE International Conference on Big Data (Big Data)*, pp. 2891–2898, IEEE, 2020.
- [49] Z. Shen and R. V. Rossel, "Automated spectroscopic modelling with optimised convolutional neural networks," *Scientific Reports*, vol. 11, no. 1, pp. 1–12, 2021.
- [50] T. Kessler, E. R. Sacia, A. T. Bell, and J. H. Mack, "Artificial neural network based predictions of cetane number for furanic biofuel additives," *Fuel*, vol. 206, pp. 171–179, 2017.
- [51] R. Piloto-Rodríguez, Y. Sánchez-Borroto, M. Lapuerta, L. Goyos-Pérez, and S. Verhelst, "Prediction of the cetane number of biodiesel using artificial neural networks and multiple linear regression," *Energy Conversion and Management*, vol. 65, pp. 255–261, 2013.
- [52] H. Yang, Z. Ring, Y. Briker, N. McLean, W. Friesen, and C. Fairbridge, "Neural network prediction of cetane number and density of diesel fuel from its chemical composition determined by lc and gc-ms," *Fuel*, vol. 81, no. 1, pp. 65–74, 2002.
- [53] K. Brudzewski, A. Kesik, K. Kołodziejczyk, U. Zborowska, and J. Ulaczyk, "Gasoline quality prediction using gas chromatography and ftir spectroscopy: An artificial intelligence approach," *Fuel*, vol. 85, no. 4, pp. 553–558, 2006.
- [54] C. Rocabrundo-Valdés, L. Ramírez-Verduzco, and J. Hernández, "Artificial neural network models to predict density, dynamic viscosity, and cetane number of biodiesel," *Fuel*, vol. 147, pp. 9–17, 2015.
- [55] I. Barra, M. Kharbach, E. M. Qannari, M. Hanafi, Y. Cherrah, and A. Bouklouze, "Predicting cetane number in diesel fuels using ftir spectroscopy and pls regression," *Vibrational Spectroscopy*, vol. 111, p. 103157, 2020.
- [56] C. H. Spiegelman, M. J. McShane, M. J. Goetz, M. Motamedi, Q. L. Yue, and G. L. Coté, "Theoretical justification of wavelength selection in pls calibration: development of a new algorithm," *Analytical chemistry*, vol. 70, no. 1, pp. 35–44, 1998.
- [57] J. A. Cramer, K. E. Kramer, K. J. Johnson, R. E. Morris, and S. L. Rose-Pehrsson, "Automated wavelength selection for spectroscopic fuel models by symmetrically contracting repeated unmoving window partial least squares," *Chemometrics and Intelligent Laboratory Systems*, vol. 92, no. 1, pp. 13–21, 2008.

- [58] X. Li, Y. Liu, X. Jiang, A. Ouyang, X. Sun, and G. Wang, "Determination and quantification of kerosene in gasoline by mid-infrared and raman spectroscopy," *Journal of Molecular Structure*, vol. 1210, p. 127760, 2020.
- [59] Z. Xiaobo, Z. Jiewen, M. J. Povey, M. Holmes, and M. Hanpin, "Variables selection methods in near-infrared spectroscopy," *Analytica chimica acta*, vol. 667, no. 1-2, pp. 14–32, 2010.
- [60] D. Jouan-Rimbaud, B. Walczak, D. Massart, I. Last, and K. Prebble, "Comparison of multivariate methods based on latent vectors and methods based on wavelength selection for the analysis of near-infrared spectroscopic data," *Analytica Chimica Acta*, vol. 304, no. 3, pp. 285–295, 1995.
- [61] T. Sennott, C. Gotianun, R. Serres, M. Ziabasharhagh, J. Mack, and R. Dibble, "Artificial neural network for predicting cetane number of biofuel candidates based on molecular structure," in *Internal Combustion Engine Division Fall Technical Conference*, vol. 56109, p. V002T02A009, American Society of Mechanical Engineers, 2013.
- [62] R. Li, J. M. Herreros, A. Tsolakis, and W. Yang, "Machine learning regression based group contribution method for cetane and octane numbers prediction of pure fuel compounds and mixtures," *Fuel*, vol. 280, p. 118589, 2020.
- [63] Y. Wang, Y. Ding, W. Wei, Y. Cao, D. F. Davidson, and R. K. Hanson, "On estimating physical and chemical properties of hydrocarbon fuels using mid-infrared ftir spectra and regularized linear models," *Fuel*, vol. 255, p. 115715, 2019.
- [64] R. M. Balabin, E. I. Lomakina, and R. Z. Safieva, "Neural network (ann) approach to biodiesel analysis: analysis of biodiesel density, kinematic viscosity, methanol and water contents using near infrared (nir) spectroscopy," *Fuel*, vol. 90, no. 5, pp. 2007–2015, 2011.
- [65] J. Zhang, X. Cui, W. Cai, and X. Shao, "A variable importance criterion for variable selection in near-infrared spectral analysis," *Science China Chemistry*, vol. 62, no. 2, pp. 271–279, 2019.
- [66] J. Zhang, X. Cui, W. Cai, and X. Shao, "Combination of heuristic optimal partner bands for variable selection in near-infrared spectral analysis," *Journal of Chemometrics*, vol. 32, no. 11, p. e2971, 2018.

- [67] T. Mehmood, K. H. Liland, L. Snipen, and S. Sæbø, "A review of variable selection methods in partial least squares regression," *Chemometrics and intelligent laboratory systems*, vol. 118, pp. 62–69, 2012.
- [68] C. M. Andersen and R. Bro, "Variable selection in regression—a tutorial," *Journal of chemometrics*, vol. 24, no. 11-12, pp. 728–737, 2010.
- [69] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and intelligent laboratory systems*, vol. 2, no. 1-3, pp. 37–52, 1987.
- [70] M. Sjöström, S. Wold, W. Lindberg, J.-Å. Persson, and H. Martens, "A multivariate calibration problem in analytical chemistry solved by partial least-squares models in latent variables," *Analytica Chimica Acta*, vol. 150, pp. 61–70, 1983.
- [71] K. Noack, B. Eskofier, J. Kiefer, C. Dilk, G. Bilow, M. Schirmer, R. Buchholz, and A. Leipertz, "Combined shifted-excitation raman difference spectroscopy and support vector regression for monitoring the algal production of complex polysaccharides," *Analyt*, vol. 138, no. 19, pp. 5639–5646, 2013.
- [72] J. C. L. Alves, C. B. Henriques, and R. J. Poppi, "Determination of diesel quality parameters using support vector regression and near infrared spectroscopy for an in-line blending optimizer system," *Fuel*, vol. 97, pp. 710–717, 2012.
- [73] G. Mendes, H. G. Aleme, and P. J. Barbeira, "Determination of octane numbers in gasoline by distillation curves and partial least squares regression," *Fuel*, vol. 97, pp. 131–136, 2012.
- [74] A. G. Abdul Jameel, V. Van Oudenhoven, A.-H. Emwas, and S. M. Sarathy, "Predicting octane number using nuclear magnetic resonance spectroscopy and artificial neural networks," *Energy & fuels*, vol. 32, no. 5, pp. 6309–6329, 2018.
- [75] R. M. Balabin and E. I. Lomakina, "Support vector machine regression (svr/lsvm)—an alternative to neural networks (ann) for analytical chemistry? comparison of nonlinear methods on near infrared (nir) spectroscopy data," *Analyst*, vol. 136, no. 8, pp. 1703–1712, 2011.
- [76] C.-Y. Wang, T.-S. Ko, and C.-C. Hsu, "Interpreting convolutional neural network for real-time volatile organic compounds detection and classification using optical emission spectroscopy of plasma," *Analytica Chimica Acta*, vol. 1179, p. 338822, 2021.



- [77] C. L. Cunha, A. S. Luna, R. C. Oliveira, G. M. Xavier, M. L. Paredes, and A. R. Torres, "Predicting the properties of biodiesel and its blends using mid-ft-ir spectroscopy and first-order multivariate calibration," *Fuel*, vol. 204, pp. 185–194, 2017.
- [78] T. Ron and R. Logeswaran, "Interpreting a neural network for stock data using lime," *J. Criti. Rev.*, vol. 7, no. 3, p. 2020, 2019.
- [79] A. Rios, V. Gala, S. Mckeever, *et al.*, "Explaining deep learning models for structured data using layer-wise relevance propagation," *arXiv preprint arXiv:2011.13429*, 2020.
- [80] R. Saluja, A. Malhi, S. Knapič, K. Främling, and C. Cavdar, "Towards a rigorous evaluation of explainability for multivariate time series," *arXiv preprint arXiv:2104.04075*, 2021.
- [81] M. C. Thrun, A. Ultsch, and L. Breuer, "Explainable ai framework for multivariate hydro-chemical time series," *Machine Learning and Knowledge Extraction*, vol. 3, no. 1, pp. 170–205, 2021.
- [82] H. Singh, A. Roy, R. Setia, and B. Pateriya, "Estimation of nitrogen content in wheat from proximal hyperspectral data using machine learning and explainable artificial intelligence (xai) approach," *Modeling Earth Systems and Environment*, pp. 1–7, 2021.
- [83] H. Taniguchi, T. Takata, M. Takechi, A. Furukawa, J. Iwasawa, A. Kawamura, T. Taniguchi, and Y. Tamura, "Explainable artificial intelligence model for diagnosis of atrial fibrillation using holter electrocardiogram waveforms," *International Heart Journal*, vol. 62, no. 3, pp. 534–539, 2021.
- [84] V. Vapnik, *The nature of statistical learning theory*. Springer science & business media, 1999.
- [85] M. Awad and R. Khanna, "Support vector regression," in *Efficient learning machines*, pp. 67–80, Springer, 2015.
- [86] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and computing*, vol. 14, no. 3, pp. 199–222, 2004.
- [87] R. Bro and A. K. Smilde, "Principal component analysis," *Analytical methods*, vol. 6, no. 9, pp. 2812–2831, 2014.

- [88] T. Hastie, R. Tibshirani, and J. Friedman, "The elements of statistical learning," *Cited on*, p. 33, 2009.
- [89] H. Abdi, "Partial least square regression (pls regression)," *Encyclopedia for research methods for the social sciences*, vol. 6, no. 4, pp. 792–795, 2003.
- [90] A.-L. Boulesteix and K. Strimmer, "Partial least squares: a versatile tool for the analysis of high-dimensional genomic data," *Briefings in bioinformatics*, vol. 8, no. 1, pp. 32–44, 2007.
- [91] B. H. Menze, B. M. Kelm, R. Masuch, U. Himmelreich, P. Bachert, W. Petrich, and F. A. Hamprecht, "A comparison of random forest and its gini importance with standard chemometric methods for the feature selection and classification of spectral data," *BMC bioinformatics*, vol. 10, no. 1, pp. 1–16, 2009.
- [92] L. Shapley, "A value for n-person games," *Ann. Math. Study* 28, *Contributions to the Theory of Games*, ed. by HW Kuhn, and AW Tucker, pp. 307–317, 1953.
- [93] P. Jaccard, "The distribution of the flora in the alpine zone. 1," *New phytologist*, vol. 11, no. 2, pp. 37–50, 1912.
- [94] M. Reza kazemi, A. Marjani, and S. Shirazian, "Development of a group contribution method based on unifac groups for the estimation of vapor pressures of pure hydrocarbon compounds," *Chemical Engineering & Technology*, vol. 36, no. 3, pp. 483–491, 2013.
- [95] S. George, "Infrared and raman characteristic group frequencies: tables and charts," Wiley, Chichester, 2001.
- [96] R. Ocampo, "Pre-built LibSVM packages for Python." <https://pypi.org/project/libsvm/>, 2022.
- [97] F. Chollet, "The Sequential model." [https://keras.io/guides/sequential\\_model/](https://keras.io/guides/sequential_model/), 2020.
- [98] Y. Goldberg and M. Elhadad, "splitsvm: fast, space-efficient, non-heuristic, polynomial kernel computation for nlp applications," in *Proceedings of ACL-08: HLT, Short Papers*, pp. 237–240, 2008.

- [99] F. Nogueira, "Bayesian Optimization: Open source constrained global optimization tool for Python." <https://github.com/fmfn/BayesianOptimization>, 2014.
- [100] J. Brownlee, "How to choose an activation function for deep learning," *Machine Learning Mastery*, 2019.
- [101] F. Chollet, "Keras BayesianOptimization Tuner." [https://keras.io/api/keras\\_tuner/tuners/bayesian/](https://keras.io/api/keras_tuner/tuners/bayesian/), 2020.
- [102] B. Liu, Y. Wei, Y. Zhang, and Q. Yang, "Deep neural networks for high dimension, low sample size data.," in *IJCAI*, pp. 2287–2293, 2017.
- [103] M. T. C. Ribeiro, "Lime: Explaining the predictions of any machine learning classifier." <https://github.com/marcotcr/lime>, 2016.
- [104] S. M. Lundberg and S.-I. Lee, "A game theoretic approach to explain the output of any machine learning model.." <https://github.com/slundberg/shap>, 2017.
- [105] H. Nori, S. Jenkins, P. Koch, and R. Caruana, "InterpretML: A Unified Framework for Machine Learning Interpretability." <https://github.com/interpretml/interpret-community>, 2019.
- [106] R. E. Bellman, *Adaptive control processes*. Princeton university press, 2015.
- [107] A. Brock, J. Donahue, and K. Simonyan, "Large scale gan training for high fidelity natural image synthesis," *arXiv preprint arXiv:1809.11096*, 2018.
- [108] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017.
- [109] T. Xu, P. Zhang, Q. Huang, H. Zhang, Z. Gan, X. Huang, and X. He, "Attngan: Fine-grained text to image generation with attentional generative adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1316–1324, 2018.
- [110] J. Kong, J. Kim, and J. Bae, "Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis," *Advances in Neural Information Processing Systems*, vol. 33, pp. 17022–17033, 2020.

- [111] M. Bińkowski, J. Donahue, S. Dieleman, A. Clark, E. Elsen, N. Casagrande, L. C. Cobo, and K. Simonyan, "High fidelity speech synthesis with adversarial networks," *arXiv preprint arXiv:1909.11646*, 2019.
- [112] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.
- [113] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.
- [114] L. Chen, S. Dai, C. Tao, H. Zhang, Z. Gan, D. Shen, Y. Zhang, G. Wang, R. Zhang, and L. Carin, "Adversarial text generation via feature-mover's distance," *Advances in Neural Information Processing Systems*, vol. 31, 2018.
- [115] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," *arXiv preprint arXiv:1710.10196*, 2017.
- [116] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125–1134, 2017.
- [117] H. Wu, S. Zheng, J. Zhang, and K. Huang, "Gp-gan: Towards realistic high-resolution image blending," in *Proceedings of the 27th ACM international conference on multimedia*, pp. 2487–2495, 2019.
- [118] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4681–4690, 2017.
- [119] Y. Dan, Y. Zhao, X. Li, S. Li, M. Hu, and J. Hu, "Generative adversarial networks (gan) based efficient sampling of chemical composition space for inverse design of inorganic materials," *npj Computational Materials*, vol. 6, no. 1, pp. 1–7, 2020.
- [120] Y. Mao, Q. He, and X. Zhao, "Designing complex architected materials with generative adversarial networks," *Science advances*, vol. 6, no. 17, p. eaaz4169, 2020.

- [121] T. Guo, D. Herber, and J. T. Allison, "Circuit synthesis using generative adversarial networks (gans)," in *AIAA Scitech 2019 Forum*, p. 2350, 2019.
- [122] L. Xu, M. Skoularidou, A. Cuesta-Infante, and K. Veeramachaneni, "Modeling tabular data using conditional gan," *Advances in Neural Information Processing Systems*, vol. 32, 2019.
- [123] D. Li, D. Chen, B. Jin, L. Shi, J. Goh, and S.-K. Ng, "Mad-gan: Multivariate anomaly detection for time series data with generative adversarial networks," in *International Conference on Artificial Neural Networks*, pp. 703–716, Springer, 2019.
- [124] J. Jordon, J. Yoon, and M. Van Der Schaar, "Pate-gan: Generating synthetic data with differential privacy guarantees," in *International conference on learning representations*, 2018.
- [125] K. Georgouli, M. T. Osorio, J. Martinez Del Rincon, and A. Koidis, "Data augmentation in food science: Synthesising spectroscopic data of vegetable oils for performance enhancement," *Journal of Chemometrics*, vol. 32, no. 6, p. e3004, 2018.
- [126] J. Houston, F. G. Glavin, and M. G. Madden, "Robust classification of high-dimensional spectroscopy data using deep learning and data synthesis," *Journal of Chemical Information and Modeling*, vol. 60, no. 4, pp. 1936–1954, 2020.
- [127] B. Yang, C. Chen, F. Chen, C. Chen, J. Tang, R. Gao, and X. Lv, "Identification of cumin and fennel from different regions based on generative adversarial networks and near infrared spectroscopy," *Spectrochimica Acta Part A: Molecular and Biomolecular Spectroscopy*, vol. 260, p. 119956, 2021.
- [128] A. Zheng, H. Yang, X. Pan, L. Yin, and Y. Feng, "Identification of multi-class drugs based on near infrared spectroscopy and bidirectional generative adversarial networks," *Sensors*, vol. 21, no. 4, p. 1088, 2021.
- [129] J. Donahue, P. Krähenbühl, and T. Darrell, "Adversarial feature learning," *arXiv preprint arXiv:1605.09782*, 2016.
- [130] G. Teng, Q. Wang, J. Kong, L. Dong, X. Cui, W. Liu, K. Wei, and W. Xiangli, "Extending the spectral database of laser-induced breakdown spectroscopy with generative adversarial nets," *Optics Express*, vol. 27, no. 5, pp. 6958–6969, 2019.

- [131] L. Zhang, Q. Nie, H. Ji, Y. Wang, Y. Wei, and D. An, "Hyperspectral imaging combined with generative adversarial network (gan)-based data augmentation to identify haploid maize kernels," *Journal of Food Composition and Analysis*, vol. 106, p. 104346, 2022.
- [132] S. Yu, H. Li, X. Li, Y. V. Fu, and F. Liu, "Classification of pathogens by raman spectroscopy combined with generative adversarial networks," *Science of The Total Environment*, vol. 726, p. 138477, 2020.
- [133] R. Yang, Y. Li, B. Qin, D. Zhao, Y. Gan, and J. Zheng, "Pesticide detection combining the wasserstein generative adversarial network and the residual neural network based on terahertz spectroscopy," *RSC Advances*, vol. 12, no. 3, pp. 1769–1776, 2022.
- [134] X. Ma, K. Wang, K. C. Chou, Q. Li, and X. Lu, "Conditional generative adversarial network for spectral recovery to accelerate single-cell raman spectroscopic analysis," *Analytical Chemistry*, 2022.
- [135] D. Zhu, L. Xu, X. Chen, L.-m. Yuan, G. Huang, L. Li, X. Chen, and W. Shi, "Synthetic spectra generated by boundary equilibrium generative adversarial networks and their applications with consensus algorithms," *Optics Express*, vol. 28, no. 12, pp. 17196–17208, 2020.
- [136] J.-J. Zhu and J. Bento, "Generative adversarial active learning," *arXiv preprint arXiv:1702.07956*, 2017.
- [137] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.
- [138] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.
- [139] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International conference on machine learning*, pp. 214–223, PMLR, 2017.
- [140] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," *Advances in neural information processing systems*, vol. 30, 2017.
- [141] Z. Lin, A. Khetan, G. Fanti, and S. Oh, "Pacgan: The power of two samples in generative adversarial networks," *Advances in neural information processing systems*, vol. 31, 2018.

- [142] A. F. Agarap, "Deep learning using rectified linear units (relu)," *arXiv preprint arXiv:1803.08375*, 2018.
- [143] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," *Advances in neural information processing systems*, vol. 29, 2016.
- [144] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," *arXiv preprint arXiv:1611.01144*, 2016.
- [145] P. Grnarova, K. Y. Levy, A. Lucchi, N. Perraudin, I. Goodfellow, T. Hofmann, and A. Krause, "A domain agnostic measure for monitoring and evaluating gans," *Advances in Neural Information Processing Systems*, vol. 32, 2019.
- [146] A. Borji, "Pros and cons of gan evaluation measures," *Computer Vision and Image Understanding*, vol. 179, pp. 41–65, 2019.
- [147] J. L. Hodges, "The significance probability of the smirnov two-sample test," *Arkiv för Matematik*, vol. 3, no. 5, pp. 469–486, 1958.
- [148] E. Härkönen, A. Hertzmann, J. Lehtinen, and S. Paris, "Ganspace: Discovering interpretable gan controls," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9841–9850, 2020.
- [149] J. S. Heyne, A. L. Boehman, and S. Kirby, "Autoignition studies of trans-and cis-decalin in an ignition quality tester (iqt) and the development of a high thermal stability uni-fuel/single battlefield fuel," *Energy & fuels*, vol. 23, no. 12, pp. 5879–5885, 2009.
- [150] A. Ratner, P. Varma, and B. Hancock, "Weak supervision: A new programming paradigm for machine learning—sail blog," *Visited on*, vol. 6, no. 26, p. 2020, 2019.