Deep Chemometrics using One Dimensional Convolutional Neural Networks for Predicting Crude Oil Properties from FTIR Spectral Data

Authors:

Souvik Ta, Shahla Alizadeh, Lakshminarayanan Samavedham, Ajay K. Ray

Date Submitted: 2022-10-19

Keywords: FTIR, Crude Oil Properties, Neural Network architectures, One Dimensional Convolutional Neural Network

Abstract:

The determination of physicochemical properties of crude oils is a very important and time-intensive process that needs elaborate laboratory procedures. Over the last few decades, several correlations have been developed to estimate these properties, but they have been very limited in their scope and range. In recent years, methods based on spectral data analysis have been shown to be very promising in characterising petroleum crude. In this work, the physicochemical properties of crude oils using FTIR spectrums are predicted. A total of 107 samples of FTIR spectral data consisting of 6840 wavenumbers is used. One Dimensional convolutional neural networks (CNNs) were used with FTIR spectral data as the one-dimensional input and Keras and TensorFlow were used for model building. The Root Mean Square Error decreased from 160 to around 60 for viscosity when compared to previous machine learning methods like PLS, PCR, and PLS-GA on the same data. The important hyper-parameters of the CNN were optimised. In addition, a comparison of results obtained with different neural network architectures is presented. Some common preprocessing techniques were also tested on the spectral data to determine their impact on model performance. To increase interpretability, the intermediate neural network layers were analysed to reveal what the convolutions represented, and sensitivity analysis was done to gather key in-sights into which wavenumbers were the most important for prediction of the crude oil properties using the neural network.

Record Type: Published Article

Submitted To: LAPSE (Living Archive for Process Systems Engineering)

Citation (overall record, always the latest version):	LAPSE:2022.0089
Citation (this specific file, latest version):	LAPSE:2022.0089-1
Citation (this specific file, this version):	LAPSE:2022.0089-1v1

CSChE Systems & Control Transactions

Open Access | Original Research

Conference Proceedings
Canadian Chemical Engineering Conference 2022
October 23-26, Vancouver, BC, Canada

PSEcommunity.org/LAPSE:2022.0089

PSE PRESS

Deep Chemometrics using One Dimensional Convolutional Neural Networks for Predicting Crude Oil Properties from FTIR Spectral Data

Souvik Ta[a], Shahla Alizadeh[a], Lakshminarayanan Samavedham[a,b] and Ajay K. Ray[a*]

[a] Department of Chemical and Biochemical Engineering, Western University, Canada N6A 5B9
[b] Department of Chemical and Biomolecular Engineering, National University of Singapore, Singapore
* Corresponding Author: aray@eng.uwo.ca

ABSTRACT

The determination of physicochemical properties of crude oils is a very important and time-intensive process that needs elaborate laboratory procedures. Over the last few decades, several correlations have been developed to estimate these properties, but they have been very limited in their scope and range. In recent years, methods based on spectral data analysis have been shown to be very promising in characterising petroleum crude. In this work, the physicochemical properties of crude oils using FTIR spectrums are predicted. A total of 107 samples of FTIR spectral data consisting of 6840 wavenumbers is used. One Dimensional convolutional neural networks (CNNs) were used with FTIR spectral data as the one-dimensional input and Keras and TensorFlow were used for model building. The Root Mean Square Error decreased from 160 to around 60 for viscosity when compared to previous machine learning methods like PLS, PCR, and PLS-GA on the same data. The important hyper-parameters of the CNN were optimised. In addition, a comparison of results obtained with different neural network architectures is presented. Some common preprocessing techniques were also tested on the spectral data to determine their impact on model performance. To increase interpretability, the intermediate neural network layers were analysed to reveal what the convolutions represented, and sensitivity analysis was done to gather key insights into which wavenumbers were the most important for prediction of the crude oil properties using the neural network.

Keywords: FTIR, Crude Oil Properties, Neural Network architectures, One Dimensional Convolutional Neural Network

Date Record: Original manuscript received October 15, 2022. Published October 19, 2022.

INTRODUCTION

Determination of crude oil properties and its characterization has long been one of the most important preliminary/critical steps for different aspects of oil refinery and reservoir calculations. Crude oil viscosity is one of the properties that determines how the crude flows in the system and hence is important to determine along with others like sulfur percentage and the cuts of other crudes. These properties are generally determined by laboratory experimentation; the related methods have evolved for years. Traditionally, the characterization of crude oil has been carried out using various chromatography methods, but these are intensive, expensive and time-consuming. The separation and identification of two or more major components could take a year or more using classical analytical methods [1]. Gas Chromatography and its combination with GC-MS has been very instrumental in analyzing petroleum components [2]. GC has been used to evaluate reservoir compartmentalization and connectivity [3]. However, these methods are extensively time and resource consuming and involve a lot of sample preparation and testing.

As an alternative, Fourier transform infrared spectroscopy technique (FTIR) has been deployed to characterize crudes. Abdulkadir et al. [1] determined that IR spectroscopy is indeed viable for characterizing crude oils and models can be using Partial Least Squares (PLS) regression on FTIR data. They used it to predict the aliphatic content and saturates for 5-7 samples. Brito et al. [4] have used human-saliva FTIR spectra coupled with support vector machines (SVMs) to find the best wavenumber regions to predict blood glucose levels.

Earlier, Principal Component Regression (PCR) on the preprocessed FTIR Spectra was used to predict density and viscosity with very good results for density and acceptable results for viscosity [5]. However, to improve the performance of the previous machine learning methods further a model with neural networks is attempted. Since fairly good performance with the PLS and PCR regression models were achieved earlier, an artificial neural network (ANN) would not provide much of a performance increase. Recently, some literature has shown

that one dimensional CNNs can be used to take spectral information as input generating good predictions of different properties [6-9] in soil, plant leaves, pharmaceutical tablets etc.

METHODOLOGY

Dataset

The crude oil samples used in this study were obtained from seven different Canadian oil fields provided by an energy corporation company in Canada. The FTIR spectral information corresponding to these samples were obtained using a Thermo Fisher Scientific FTIR microscope. Other physicochemical properties crucial to the characterization of crude were obtained using appropriate analytical instruments and laboratory methods by the company was provided. Overall, 107 samples of crude with 6366 wavenumbers each are used as attributes. The output space had 13 properties with most important being density and viscosity. Other properties included sulphur content, Total Acid Number, Micro Carbon Residue and the yields of different cuts of the crude.

Dataset Analysis and Preprocessing

Upon analyzing the correlation of output space attributes, it was found that most of them were highly correlated with density or viscosity. In our previous paper [5], both density and viscosity were predicted using machine learning methods; particularly good predictions were obtained for density using PCR, but the error for viscosity was not within acceptable limits. Hence, the more advanced CNN architecture for predicting the viscosity was investigated. The data was cleaned by removing the first 476 wavenumbers from the spectra due to missing values and noise. The resulting data was subject to auto-scaling so that all variables (spectral values at each wavenumber) have mean equal to zero and standard deviation equal to one.

Convolutional Neural Networks

CNN is a widely used class of deep learning architectures primarily used in computer vision applications. Recent studies have started to use a modified CNN called a one-dimensional CNN to predict properties from spectral information [6-9]. In these applications, the spectral information is regarded as a one-dimensional image and fed to the input layer. CNNs are generally made up of three types of layers: convolutional layers, pooling layers and fully connected layers and a network is created by stacking them [10].

Convolutional Layer

The convolutional layer is the most vital layer in a CNN and uses learnable kernels to train. The filters in a convolutional layer convolve over the entire input to generate a 2D activation map [10, 11]. The network then learns the values in the kernel to fire when a specific feature is detected. These layers can decrease the complexity and number of parameters compared to a traditional ANN. The common hyper-parameters for this layer are depth, stride and zero-padding. Depth is the depth of the output or the number of kernels; Stride is the number of pixels the filters move by when convolving on the input. Increasing stride can result in less overlap and reduce the output dimensions but it can also capture less data. Zero-padding is used to pad the border of the input with zeros and hence preserve the data near the corners and also the dimensions.

To calculate the output dimensions of the convolutional layer, the formula used is Output Size = 1+ (N-F)/S

where N is Input Size, F is Filter Size and S is Stride,

Pooling layer

Pooling layers are used to reduce the dimensions of the layers which further reduces the number of parameters and the computational complexity, The most common kind of pooling is max pooling which replaces the value of a kernel with the MAX value inside it [11]. The most common filters used for max pooling are 2*2 with a stride of 2. This doesn't cause any overlap of filters. Generally, increasing the kernel size causes a loss in information and decreases performance greatly [10].

Fully Connected Layer

A fully connected layer is very similar to how neurons are connected in a traditional ANN in which each neuron in one layer is connected to each neuron in the next layer. This generally results in a lot of trainable parameters and is generally used to connect the features from the convolutions to the output [10, 11].

Common architectures

The CNN's generally follow a common architecture in which one cant just connect any type of layer after another. Generally convolutional layers are stacked with pooling layers and this forms an unit of a convolutional layer and a pooling layer which is repeated and is finished with a fully connected layer. Sometimes stacking two convolutional layers followed by a pooling layer to form an unit helps in selecting more complex features [10].

Gaussian Noise

A Gaussian noise layer was used with a standard deviation of 0.05. This layer adds noise with a mean of zero and a specified standard deviation to the input layer. Studies [12] suggest adding Gaussian Noise can have a regularizing effect and reduce overfitting.

Dropout Layer

Dropout randomly drops nodes from the layer while training and simulates the effect of ensemble learning. In this work, a dropout layer was used to add a regularizing effect and prevent overfitting. The amount of dropout to use will be investigated as a hyperparameter [13].

One Dimensional CNNs

One dimensional CNNs are a modified version of the conventional deep CNNs and use only an one dimensional input of the shape [n,1]. 1-D CNNs have shown some advantages over the deeper traditional CNNs [14]. They require simple array operations as opposed to more bulky matrix operations in CNNs; this significantly reduces the computational load. Studies have shown that 1D CNNs are really good at performing signal processing tasks with a relatively shallower architecture which is easier to train and implement. Since we are using this in an industrial setting, 1D CNNs having a much lower computational load helps in implementing it as a more cost effective and real time solution.

1D CNNs have been used in a lot of signal processing applications including those that involve ECGs signal and vibration data[14]. In recent studies, 1D CNNs have been proved to work very well with spectral information as an input

– this enabled gaining insights and useful predictions from data. There have been applications in soil quality predictions [6, 7]. Kawamura et al. [7] have used it on Vis-NIR spectra to estimate available phosphorus in soil. Ng et al. [6] have also used spectral information from combined sources like Vis-NIR and MIR to predict several soil properties. Prilianti et al. [8] have successfully used a 1D CNN to predict pigment concentration in leafs from the reflectance spectra. Bjerrum et al. [9]have developed methods like data augmentation and scatter correction specially for 1DCNNs and successfully predicted drug content in tablets from NIR spectra. However, there is lack of literature that describes the use of 1D CNNs in petrochemistry with FTIR spectra and this is the matter of investigation in this study.

Hyperparameters

The most important hyperparameters to be tuned were the batch size, learning rate and the optimization technique employed. All models were trained with epochs around 200-300 with early stopping and reduceLRonPlateau callbacks.

Table 1: Initial Hyperparameter Search Space

Hyperparameter	Search Space
Batch size	8,16,32,64
Epochs	150-300
Learning Rate	0.01,0.001

We tested seven different gradient descent-based optimization techniques to train our algorithms. They are Stochastic Gradient Descent, Adagrad, Adadelta, RMSProp, Adam, Adamax and Nadam [15].

Neural Network Architecture

Previous studies of using 1D CNNs on spectra suggest that shallow networks perform much better than deep networks and hence we decided to go forth with shallow networks for our models. This also helps us with the reduced computational load and aids realtime application[8]. For the initial testing of hyperparameters, a standard 2 hidden layer CNN was used alongwith Gaussian Noise and Dropout. The architecture is shown in Table 2.

Table 2: Architecture of 1D CNN used for hyperparameter testing.

Layer	Parameters
Gaussian Noise	Standard deviation = 0.05
Conv 1D	no of filters = 32; kernel size = 8; ReLU
Conv 1D	no of filters = 32; kernel size = 16; ReLU
Dropout	dropout = 0.5
Dense Layer	no of units = 128; ReLU
Dense Layer	no of units = 1; Linear

After the initial hyperparameters were determined, the best batch size and learning rate were fixed to decide the best neural network architecture for our data. For this, a Neural Architecture Search approach was implemented using kerastuner in Tensorflow [16]. Neural Architecture Search is automated architecture engineering algorithm which determines the best neural network architecture. Generally, it has three dimensions: Search Space, Search Strategy and Performance Estimation Strategy [17].

Search Space: This consists of the possible neural architectures the algorithm will search through. In our study, we consider shallow networks only. For the initial runs we considered two convolutional layers and one fully connected layer with the hyperparameters as the number of kernels and filter size for each layer and the number of dense connected units. Later, we increased this and the algorithm had to decide between 2 to 5 hidden layers and the number of kernels and filter size in each.

Search Strategy: This deals with the algorithm to navigate the search space. The most popular strategies are Random, Bayesian Optimization [18] and HyperBand Tuner[19].For our study, we decided to use Bayesian Optimization since it would be too computationally expensive to do an exhaustive search or a RandomSearch.

Performance Estimation Strategy: A normal Train-Test validation was investigated since we have a small dataset.

RESULTS AND DISCUSSION

All the hyperparameters were first tested with a standard 1D CNN containing 2 hidden layers and 2 fully connected layers to get a good estimate of hyperparameters to do the neural architecture search.

Hyperparameters

Batch size

The best batchsize tends to depend a lot on the learning rate and strongly on the optimization algorithm used but a general trend was the the error increased as the batch size increased. For the majority of test cases a batchsize of 8 or 16 gave the best results.

Epochs

The best epoch size was found manually from the loss graphs.Most of the models were found to converge in between 200 to 300 epochs so an epoch count of 300 was chosen.

Learning rate

For learning rate we implemented a function which reduced learning rate during training if the loss plateaued for a certain number of epochs. Two learning rates of 0.01 and 0.001 were tested and the learning rate would become half if the loss was stagnant for 10 epochs or more. For all test cases a lower learning rate of 0.001 gave better results and the learning rate plateau function was triggered frequently.

Optimization algorithms

All the optimization algorithms were run twice for different sets of the above hyperparameters. SGD appeared to be very

unstable and did not converge at all. All the optimization algorithms had much better performance at a lower learning rate of 0.001 than 0.01 except Adagrad which seemed to be an exception on repeated testing. All the algorithms performed best at batchsize = 8 except Nadam which gave pretty even performance for both 8 and 16.

Adam, Adamax and Nadam had the best values of RMSE at learning rate equal to 0.001 with Nadam being the most consistent through its runs. RMSProp gave comparable RMSE values but was very unstable and gave very different values of error for each run. Finally due to its stability across different runs and low RMSE values we decided upon Nadam as the preferred optimization algorithm with a learning rate of 0.001 and a batchsize of 8 or 16.

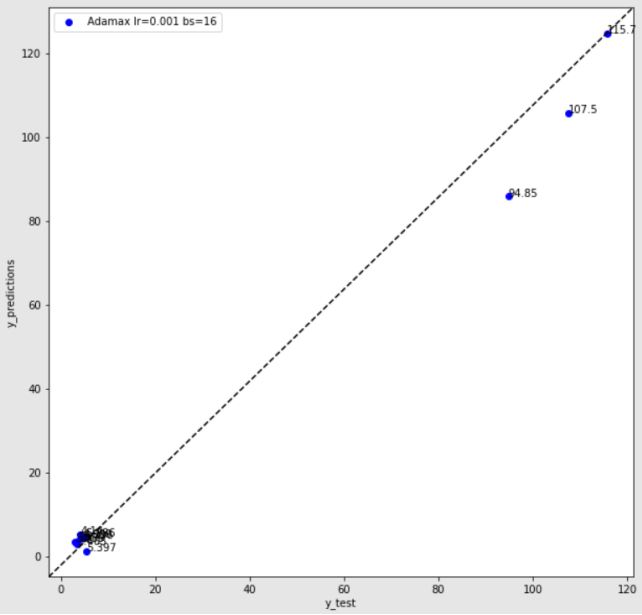

Figure 1. predicted viscosity vs actual viscosity for standard neural network with Adamax,learning rate=0.001,batchsize=16

Table 3: Results of initial hyperparameter optimization. [RMSE values on right; where lr: learning rate, bs: batch size.]

Optimization Algorithm	$lr=10^{-2}$ bs=8	$lr=10^{-2}$ bs=16	$lr=10^{-3}$ bs=8	$lr=10^{-3}$ bs=16
Adadelta	70	83	NA	NA
RMSProp	201.68	207.89	33.67	22.474
Adagrad	80.964	75.736	333.67	132.11
Adam	2349.1	2349.1	30.408	30.527
Adamax	105.75	2827.6	114.96	5.59
Nadam	150.72	252.11	5	22

Neural Architecture Search

Since very low RMSE values of around 5 to 20 was achieved during our initial hyperparameter process with just two layers, there were concerns of overfit if the number of layers were increased and hence the search space was limited to 2 hidden layers and one dense layer. The search space is as follows.

Table 4: Architecture of 1D CNN used for hyperparameter testing.

Hyperparameter	Search Space
layer1_filters	[32,64,96,128...256] step=32
layer1_kernels	[2,4,8,16,24,32]
layer2_filters	[32,64,96,128...256] step=32
layer2_kernels	[2,4,8,16,24,32]
Dense Layer	[32,48,64,80,96,112,128]

An exhaustive search of all the possible combinations of the architecture would be too computationally expensive so Bayesian Optimization for the search was used.

The best results are as follows:

Table 5: Results of hyperparameter optimization of the neural network using Bayesian Neural Architecture Search

layer1_filters	layer1_kernels	layer2_filters	layer2_kernels	Dense	RMSE
32	16	256	8	128	6.12
32	16	224	2	128	9.45
32	12	256	4	128	16.5
32	32	256	2	96	7.2
32	32	256	2	64	8.5
32	32	256	2	80	13.6
32	32	256	2	48	28.6

Multiple runs of Bayesian Optimization was run and the best values for filters of layer 1 and 2 were 32 and 256. From the results we can see layer 1 preferred to be the minimum value of 32 consistently while for layer 2 the number of filters was always in the higher range above 200. The results are in Table 5.

For Kernels we see more variation but the trends suggest that layer 1 kernels prefer to be at the maximum end around 16 and layer 2 kernels prefer to be in the lower range mostly 2,4 and 8.

A run of Bayesian Optimization was done to confirm this where the layer1 and layer2 filters were fixed at 32 and 256. The results showed that layer 1 preferred to be a higher value of 32 and layer 2 stayed around 2 or 4. These results are in Table 6.

A run of Bayesian Optimization was done without Gaussian Noise and it gave much worse average performance than previous runs and the trend of hyperparameters were inconsistent thus showing that Gaussian Noise is important.

Table 6: Results of optimization of the number of kernels using Bayesian Neural Architecture Search

l1_kernel	l2_kernel	RMSE
32	4	5.86
32	2	10.629
16	2	66.776
2	8	70.496

ACKNOWLEDGEMENTS

The project is funded by Natural Sciences and Engineering Research Council under Alliance Grants number R3839A44.

REFERENCES

1. Abdulkadir I, Uba S, Almustapha MN. A Rapid Method of Crude Oil Analysis Using FT-IR Spectroscopy. *Nig. J. Basic Appl. Sci.* 24(1):47-55 (2016).
2. Speight J, Handbook of Petroleum Product Analysis: Second Edition (2015).
3. Larter SR, Aplin AC. Reservoir geochemistry: methods, applications and opportunities. *Geol. Soc. Lond.* 86(1):5–32 (1995).
4. Sánchez-Brito M, Luna-Rosas FJ, Mendoza-González R, Mata-Miranda MM, Martínez-Romo JC, Vázquez-Zapién GJ. A machine-learning strategy to evaluate the use of FTIR spectra of saliva for a good control of type 2 diabetes. *Talanta* 221:121650 (2021).
5. Alizadeh S, Ta S, Ray AK, Lakshminarayanan S. Determination of Density and Viscosity of Crude Oil Samples from FTIR Data using Multivariate Regression, Variable Selection and Classification. *IFAC-PapersOnLine* 55(7):845–850 (2022).
6. Ng W, Minasny B, Montazerolghaem M, Padarian J, Ferguson R, Bailey S, McBratney AB. Convolutional neural network for simultaneous prediction of several soil properties using visible/near-infrared, mid-infrared, and their combined spectra. *Geoderma* 352:251–267 (2019).
7. Kawamura K, Nishigaki T, Andriamananjara A, Rakotonindrina H, Tsujimoto Y, Moritsuka N, Rabenarivo M, Razafimbelo T. Using a one-dimensional convolutional neural network on visible and near-infrared spectroscopy to improve soil phosphorus prediction in Madagascar. *Remote Sens* 13(8):1519 (2021).
8. Prilianti KR, Setiyono E, Kelana OH, Brotosudarmo THP. Deep chemometrics for nondestructive photosynthetic pigments prediction using leaf reflectance spectra. *Inf. Process. Agric.* 8(1):194–204 (2021).
9. Bjerrum EJ, Glahder M, Skov T. Data Augmentation of Spectral Data for Convolutional Neural Network (CNN) Based Deep Chemometrics. *arXiv preprint* arXiv:1710.01927 (2017).
10. O'Shea K, Nash R. An Introduction to Convolutional Neural Networks. *arXiv preprint* arXiv:1511.08458 (2015).
11. Albawi S, Mohammed TA, Al-Zawi S. Understanding of a convolutional neural network. in *Proceedings of 2017 International Conference on Engineering and Technology (ICET) 1-6 (2017)*.
12. Bjerrum EJ, Glahder M, Skov T. Data Augmentation of Spectral Data for Convolutional Neural Network (CNN) Based Deep Chemometrics. *arXiv preprint* arXiv:1710.01927 (2017).
13. Srivastava N, Hinton G, Krizhevsky A, Salakhutdinov R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.* 15(1):1929-1958 (2014).
14. Kiranyaz S, Avci O, Abdeljaber O, Ince T, Gabbouj M, Inman DJ. D Convolutional Neural Networks and Applications-A Survey. *Mech. Syst. Signal Process.* 151:107398 (2021).
15. Ruder S. An overview of gradient descent optimization algorithms. *arXiv preprint* arXiv:1609.04747 (2016).
16. Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, Corrado GS, Davis A, Dean J, Devin M, Ghemawat S. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *arXiv preprint* arXiv:1603.04467 (2016).
17. Elsken T, Metzen JH, Hutter F. Neural Architecture Search: A Survey. *J. Mach. Learn. Res.* 20(1):1997-2017 (2019).
18. Kandasamy K, Neiswanger W, Schneider J, Poczos B, Xing E. Neural Architecture Search with Bayesian Optimisation and Optimal Transport. *Adv. Neural Inf. Process. Syst.* 31 (2018).
19. Li L, Jamieson K, Rostamizadeh A, Talwalkar A. Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization. *J. Mach. Learn. Res.* 18(1):6765-6816 (2017).

This conference proceeding has not been peer reviewed.