Neural Computing and Applications (2023) 35:18719–18735
https://doi.org/10.1007/s00521-023-08700-z

ORIGINAL ARTICLE

RamanNet: a generalized neural network architecture for Raman spectrum analysis

Nabil Ibtehaz[1] · Muhammad E. H. Chowdhury[2] · Amith Khandakar[2] · Serkan Kiranyaz[2] · M. Sohel Rahman[3] · Susu M. Zughaier[4]

Received: 16 June 2022 / Accepted: 23 May 2023 / Published online: 21 June 2023

Abstract
Raman spectroscopy provides a vibrational profile of the molecules and thus can be used to uniquely identify different kinds of materials. This sort of molecule fingerprinting has thus led to the widespread application of Raman spectrum in various fields like medical diagnosis, forensics, mineralogy, bacteriology, virology, etc. Despite the recent rise in Raman spectra data volume, there has not been any significant effort in developing generalized machine learning methods targeted toward Raman spectra analysis. We examine, experiment, and evaluate existing methods and conjecture that neither current sequential models nor traditional machine learning models are satisfactorily sufficient to analyze Raman spectra. Both have their perks and pitfalls; therefore, we attempt to mix the best of both worlds and propose a novel network architecture RamanNet. RamanNet is immune to the invariance property in convolutional neural networks (CNNs) and at the same time better than traditional machine learning models for the inclusion of sparse connectivity. This has been achieved by incorporating shifted multi-layer perceptrons (MLP) at the earlier levels of the network to extract significant features across the entire spectrum, which are further refined by the inclusion of triplet loss in the hidden layers. Our experiments on 4 public datasets demonstrate superior performance over the much more complex state-of-the-art methods, and thus, RamanNet has the potential to become the de facto standard in Raman spectra data analysis.

Keywords Raman spectrum analysis · Convolutional Neural Networks · Multilayer perceptron · Deep learning · Neural network

1 Introduction

Raman scattering is one of the various light-matter interactions, comprising absorption and subsequent emission of light by matter [1]. Spectroscopy, being the study of the interaction of light or broader electromagnetic radiation with matter, thus has also focused on Raman scattering, ever since its discovery in 1928 by Raman and Krishnan [2]. Unlike elastic scattering, e.g., Raleigh scattering, the wavelength of incident light changes in Raman scattering [3]. Following the typical norm of inelastic light scattering, when a photon excites the sample, the electrons are raised to a higher virtual energy state [4]. This excitation event is usually short-lived and the molecule soon reaches a new stable energy state, either lower (Stokes shift) or higher (anti-Stokes) [5]. Based on the difference in energy, the sample achieves a different vibrational and rotational state. Therefore, Raman spectroscopy can be used to analyze the vibrational modes of various molecules, extracting the structural fingerprint of such materials in the process [1].

Being capable of uniquely fingerprinting materials, Raman spectra have been used in a wide variety of applications [6], covering medical diagnosis [7], forensics [8], mineralogy [9], bacteriology [10], virology [11], etc.

Conventionally, Raman spectra are analyzed in terms of wavenumber $\tilde{\nu}$ (cm^{-1}). The standard practice is to present them with the wavenumber shift, linearly increasing along the horizontal axis. On the contrary, the vertical axis ordinate is proportional to intensity [5]. Therefore, this setting does not appear too different from traditional spectrograms. However, the issue in treating Raman spectra as typical spectrograms is that for Raman spectrum, the horizontal axis does not represent time. Thus, it is not logically sound to apply models that are used for spectrum

Extended author information available on the last page of the article

Table 1 Hyperparameters of RamanNet

Hyperparameter	w	dw	n_1	n_2	n_f	dp_1	dp_2	dp_3
Value	50	25	25	512	256	0.5	0.4	0.25

analysis, e.g., convolutional neural networks (CNN), as they discard the (time-domain) locality of the spectrum, which is crucial for Raman spectra. On the contrary, although the traditional machine learning methods, such as support vector machine or logistic regression are more suitable to deal with Raman spectra, they suffer from the curse of dimensionality as Raman spectra data is usually long. Principal Component Analysis (PCA) has been widely used for feature reduction, as 34 out of recent 52 papers used that [6]. Still, it may not always be able to compress Raman spectra properly, as PCA is more suitable for tabular data sources, where the features are ideally uncorrelated, but in Raman spectra, the intensities at nearby Raman shifts are expected to demonstrate some sort of correlation.

With the reduction of cost and complexity related to the Raman data extraction pipeline, there has been an unprecedented expansion in Raman datasets in recent years. This sudden growth of available Raman data requires suitable methods to analyze them properly. Deep learning is achieving state-of-the-art results in multiple domains including natural language processing [38], computer vision [34], healthcare [39], education [37], psychology [36] etc. However, to the best of our knowledge, there has not been any deep learning methodology, devised solely focusing on Raman spectra data, considering the pattern and properties of this unique data source. To this end, we carefully contemplate the properties of Raman spectra data and corresponding attributes of contemporary machine learning methods. We argue why the application of convolutional neural networks (CNN) may not be appropriate for the invariance properties, but at the same time acknowledge that CNN is more capable than traditional machine learning methods due to the sparse connection and weight-sharing properties. We make an attempt to fuse the best of both worlds and propose RamanNet, a deep learning model that also follows the paradigm of sparse connectivity without the limitation of temporal or spatial invariance. We achieve this by using shifted densely connected layers [35] and emulate sparse connectivity found in CNN, without the concern of invariance. Furthermore, dimensionality reduction has been involved in most Raman spectra applications, and thus we employ triplet loss in our hidden layers and make more separable and informative embeddings in lower-dimensional space. Our experiments on 4 public datasets demonstrate superior performance over the much more complex state-of-the-art methods, and thus, RamanNet has the potential to become the de facto standard in Raman spectra data analysis.

In summary, we have attempted to generalize the task of Raman spectrum analysis using a novel neural network architecture, RamanNet. The primary contributions of this work include:

- We leveraged densely connected layers for feature extraction instead of CNN layers, as the Raman spectrum is different from typical temporal spectra.
- We further made the computations efficient by incorporating sparse connectivity similar to CNNs, at the same time preventing temporal invariance.
- Raman spectra being substantially long, dimensionality reduction is an important application. To this end, we incorporated triplet loss in the hidden layers so that the reduced feature maps become more meaningful.
- We performed a thorough evaluation of the proposed RamanNet architecture with current state-of-the-art methods on 4 different and unrelated datasets.
- Additionally, we inspected the learned feature maps and attempted to interpret them.

2 Motivations and high-level considerations

From a visual perception, the Raman spectrum resembles a signal-like waveform, which has led to the application of one-dimensional (1D) convolutional neural networks to analyze Raman spectra [9, 12, 13]. However, the Raman spectrum is not a typical signal rather it is an energy distribution plot, which may not be suitable for the appropriate utilization of CNNs as the subsequent discussion unfolds. In what follows, we briefly discuss the properties of CNNs and present our motivations and rationale.

Convolutional neural networks are one of the most successful and widely used neural network architectures, particularly in computer vision [14] and signal processing [15] domains. The success of CNNs can be largely attributed to three primary properties of CNNs, namely sparse interaction, parameter sharing, and equivariant representations [16]. The nature of image or signal data, i.e., spatial or temporal properties, works in perfect harmony with the properties of CNN. Unlike traditional feed-forward neural networks, where all the input features are connected with all the hidden or output layers, CNNs employ sparse connectivity by leveraging kernels. Only a small portion of the input is analyzed at a specific step using kernels, and thus this prevents the computation from being overwhelmed with analyzing the entire input at once. Although the focus is put on local information, global information is also

considered by appropriate use of pooling operations along with the broader field of vision in deeper networks. Furthermore, the use of kernel-based computation also facilitates parameter sharing, reducing the number of parameters and the risk of overfitting simultaneously. Another vital property of CNN is the equivariance in representation. Mathematically for a convolutional operation f on an image I, $f(I(x,y)) = f(I(x-h, y-k))$, i.e., a point of interest whether it resides at position (x, y) or at a shifted position $(x-h, y-k)$, the output of the convolution operation remains the same.

This equivariance to translation plays a pivotal role in working with image or signal data. When processing signal data, this property implies that CNN generates a timeline of the emergence of different key points in the signal. As a result, regardless of the time of occurrence, all the features in a signal are captured, unlike traditional neural networks which would have only sought the features at the exactly fixed timestamps. Furthermore, the application of global pooling operations makes sure that all the feature signatures are preserved in the final representation.

In Fig. 1, a simplified example of how CNN works with a signal as input has been presented. In Fig. 1a, there are three shifted ECG signals; since the time of occurrence of a particular feature in a signal is uncertain, it is imperative that the model can identify the feature, irrespective of the timestamp. In Fig. 1b, it can be observed that the feature maps from a convolutional layer and apparent that the equivalent features are detected, albeit shifted in accordance with the shifted nature of the signals, complying with the property $f(I(x,y)) = f(I(x-h, y-k))$. Finally, a global pooling operation summarizes the feature maps and provides identical representations of all the three signals in Fig. 1c.

The example clearly shows the suitability of applying CNNs for the time-series signal data. Since the pattern of Raman spectrum closely resembles the pattern of signal data, it is trivial to use CNNs to analyze Raman spectra. The sparse connectivity and parameter sharing truly help in this regard, as it makes the computation simpler and less prone to overfitting. However, the equivariance to translation property which proved vital for analyzing signals is not useful when working with the Raman spectra as follows. The Raman spectrum is plotted as intensity vs Raman shifts. Thus the concept of equivariant translation is not applicable in this case, because similar intensity at different Raman shifts implies a completely different meaning. For example, in Fig. 2, we present a similar scenario (as in Fig. 1) but with the Raman spectrum as input. It can be seen that we have three completely different Raman spectrums, having similar patterns in different Raman shifts. But the CNN model, due to the translational equivariance, treats them as shifted inputs and generates the same output for all of them (Fig. 2c).

The above examples resurface the question of the suitability and efficacy of CNN in Raman spectrum analysis. Attempts have been made to address this question in prior works by using deeper networks (broadening the field of vision thereby) along with discarding pooling layers (with a goal to preserve the locations of spectral peaks) [12]. On the contrary, we can use traditional neural networks or machine learning models which are capable of keeping track of the exact locations. However, they succumb to the curse of dimensionality [17], due to the long length of the Raman spectrum. Although PCA has been mostly used to reduce that [6], still it feels less intuitive to model spectrum inputs using hyperplane optimization, which is more suitable for tabular data. As the spectra are apparently correlated in neighboring Raman shifts, this rather motivates us to use sparse connectivity through kernel-like operations in CNN instead, which is also supported by the reduced accuracy in classical models [9, 12].

This brings us to the dilemma of whether to use CNN or classical machine learning pipelines. On one hand, CNNs

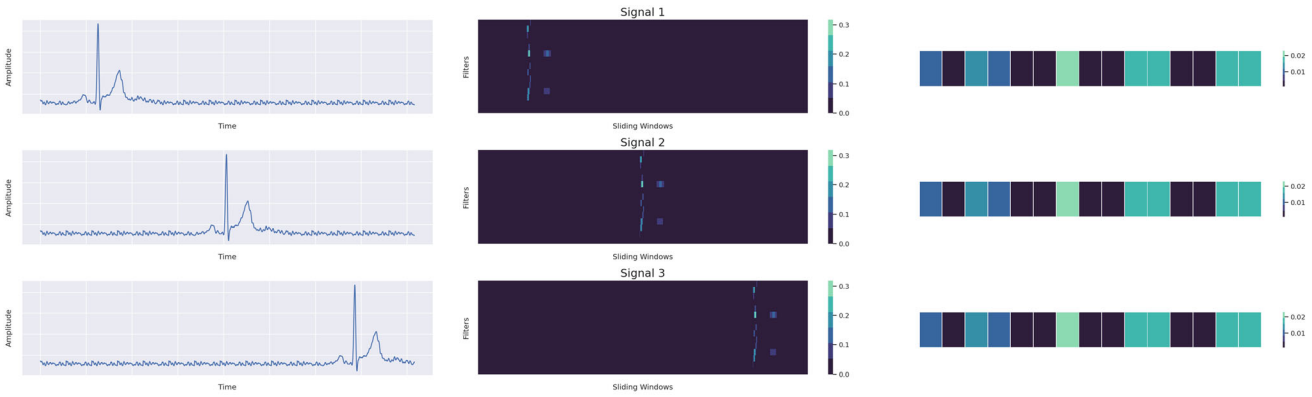


Fig. 1 Analysis of CNN for a 1-D time-series signal input

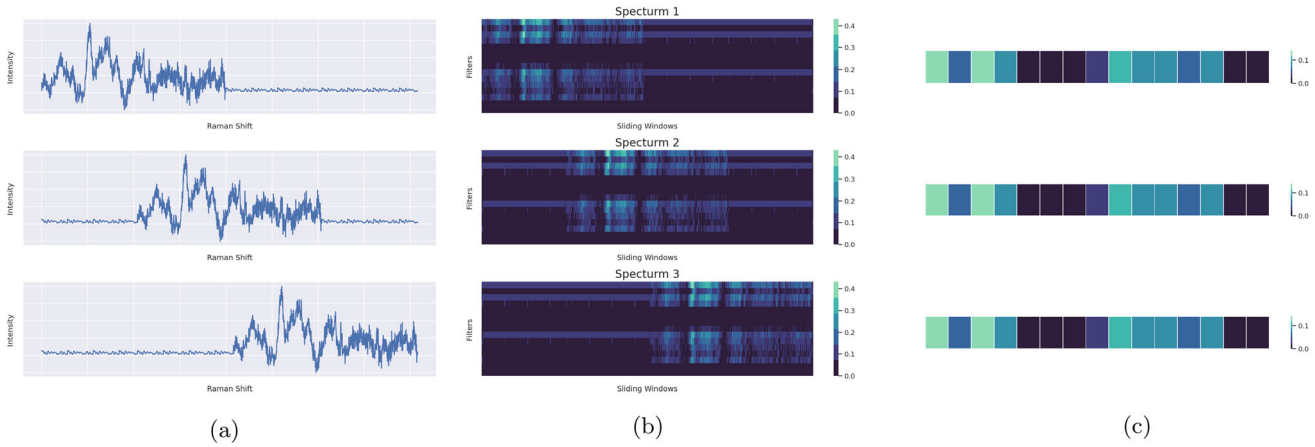


Fig. 2 Analysis of CNN for a Raman spectrum input

help in analyzing Raman spectra for the benefits of sparse connectivity (considering the correlation in neighboring Raman shifts) and parameter sharing (reducing the number of parameters and overfitting), but lose their applicability due to translational equivariance. On the other hand, traditional models are free from translational equivariance, but they cannot comprehend the correlation in the neighboring Raman shifts and suffers from the curse of dimensionality as trying to optimize too many parameters at once. In the sequel, we overcome this dilemma by proposing a middle ground between the two approaches, fusing the best of the both worlds. We propose to use shifted multi-layer perceptrons (MLPs) [35] to analyze shifted windows of Raman spectra. This facilitates sparse connectivity as the shifted MLP layers mimic kernels and only analyze a part of the input. Moreover, parameter sharing is redundant in this regard, as the kernel-like operations of a particular kernel are only performed in one part and not elsewhere. Finally, this also eliminates the issues with translation equivarience, as for different locations or different shifted windows we have different MLP layers. Thus,

Mathematically, a typical 1D CNN operation can be simplified as,

$$y(i) = \sigma\left(\sum_h x(i+h)k(h) + b\right) \tag{1}$$

Here, x is the 1D input, y is the output, k is a learned kernel, b is the bias term and σ is a nonlinear operation. The same kernel k is applied everywhere, and thus the translational equivarience is achieved.

On the contrary, our proposed modification from using an MLP,

$$y(i) = \sigma(W_{f(i)}^T x + b) \equiv \sigma\left(\sum_h x(i+h)k_{f(i)}(h) + b\right) \tag{2}$$

Here, the dot product $W^T x$ is mathematically equivalent to a 1D convolutional operation with proper relation between the weight matrix W and kernel k. In addition, since we are using sliding windows, the weight matrix $W_{f(i)}$ and kernel $k_{f(i)}$ depends on the location, i.e., the value of i, and this relation is represented by the function $f(i)$.

3 Proposed architecture

On the basis of the above discussion, a novel network architecture, RamanNet is presented here. As mentioned in the previous section, the convolution operation is mimicked using multi-layer perceptrons or so-called densely connected neural network layers. The input Raman spectrum is broken into overlapping sliding windows of length w and step size dw, and each of them is passed to a different dense block with n_1 neurons each. This ensures sparse connectivity and reduces the risk of overfitting. Furthermore, this configuration somewhat resembles a convolution operation without translation, and thus the features extracted at the neurons can be considered the same way as features learned from kernels would be considered.

The features from all the dense blocks are concatenated together, and a dropout dp_1 is applied. These concatenated features are again summarized using another dense layer with n_2 neurons. The outputs from the summarization are regularized with a dropout of dp_2.

Finally, n_f features are computed from the regularized outputs using another dense layer with n_f neurons. This layer is named as embedding layer. Raman spectra are known to be noisy with a low signal-to-noise ratio (SNR) [12], which often leads to less separation between the classes. Therefore, in order to improve this, triplet loss [18] is introduced as an auxiliary loss. Triplet loss is defined

using Euclidian distance, f, between an anchor A, positive example P and negative example N as,

$$L(A,P,N) = max(||f(A) - f(P)||^2 - ||f(A) - f(N)||^2 + \alpha, 0) \quad (3)$$

Here, α is a margin term to ensure that the model learns non-trivial information.

Using triplet loss, a well-separated embedding space can be obtained. The embeddings are finally used to predict the classes of input using the Softmax activation function and a dropout of dp_3. The network is thus updated using a combination of triplet and cross-entropy loss.

$$Loss = 0.5 \times \text{triplet loss} + 0.5 \times \text{cross-entropy loss} \quad (4)$$

Other than the output layer, all the layers use LeakyRELU activation function [19] and are batch normalized [20]. The values of the various hyperparameters are as follows:

A simplified diagram of the RamanNet is presented in Fig. 3.

4 Datasets

One particular limitation when working with deep learning architectures for Raman spectrum analysis is the lack of sufficient public benchmark datasets [6]. Although there have been several recent works utilizing deep learning models for Raman spectrum analysis, the datasets are mostly proprietary or private [21, 22]. From an elaborate review of the recent works on Raman Spectrum analysis [12, 13, 24, 25], 4 publicly available datasets were selected for analyzing and benchmarking RamanNet. It would have been preferable to include more datasets in our evaluation, but unfortunately at the time of writing this paper, there were no more additional Raman spectra databases suitable for training machine learning methods. Nevertheless, it should be noted that the majority of the existing methods were evaluated on only one or two similar datasets.

4.1 COVID dataset

We used the publicly available data [23] from the recent work [24], which acts as a pilot study of primary screening of COVID-19 (COronaVIrus Disease 2019) by Raman spectroscopy. This dataset contains a total of 177 serum samples collected from 63 COVID-19 patients, 59 suspected ones, and 55 healthy people (i.e., the control group). The COVID-19 group was recruited at the Chengdu Public Health Clinical Medical Center, and it includes 58 symptomatic and 5 asymptomatic patients. The suspected group demonstrated flu-like symptoms but tested negative using RT-PCR tests. For all the subjects, 1-hour repose of blood sampling the serum was extracted by centrifuging at 3000 *rpm* for 10 min and was stored at 4 °C. Later, a single-mode laser diode with 785 *nm* wavelength and 100 *mW* power was used for Raman excitation. The laser power applied to the sample was measured at around 70 *mW*, and the spectra were recorded in the range of 600–1800 cm^{-1}.

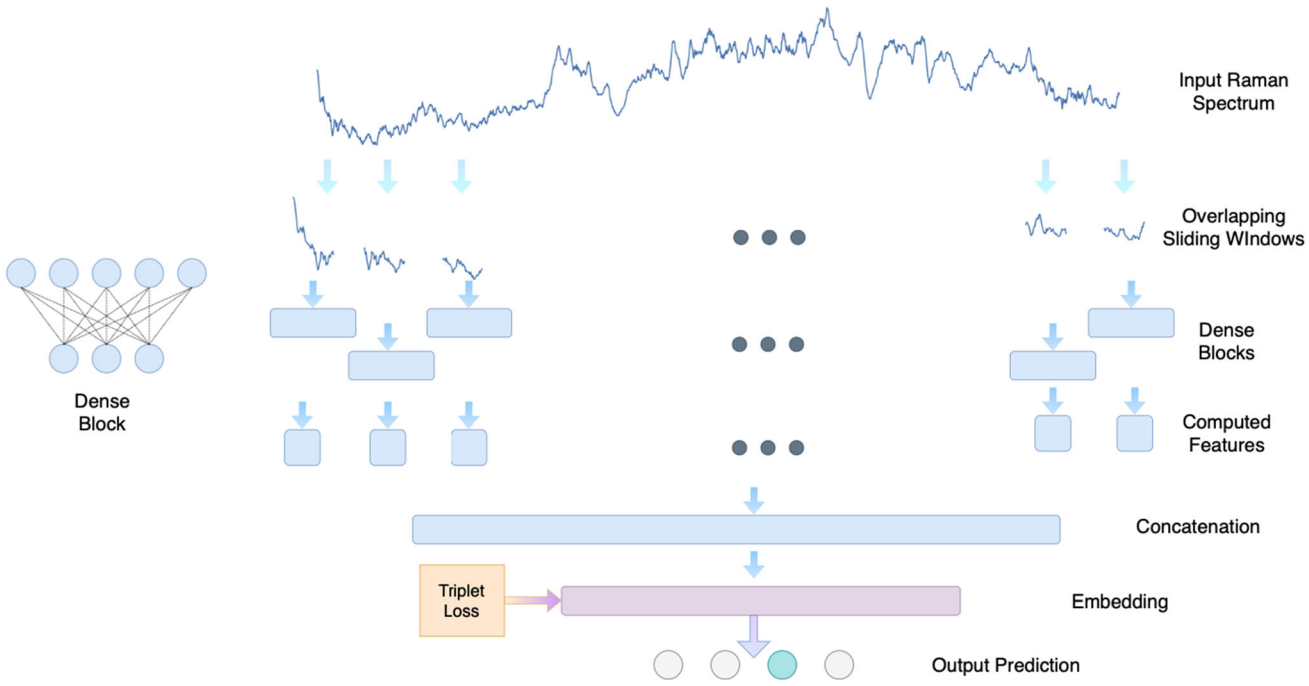


Fig. 3 RamanNet Architecture

4.2 Melanoma dataset

Erzina et al. [13] used surface enhanced Raman spectroscopy (SERS) to detect skin melanoma. Healthy residual skin and skin melanoma metastasis were collected. From cell line and primary culture, 12 categories of samples were considered, each having 8–9 samples. The SERS spectra were measured using a ProRaman-L spectrometer at 785 *nm* excitation wavelength and 33 *mW* power. AuMs, functionalized by ADT–NH_2, ADT–COOH, or ADT$-(COOH)_2$ were used to measure the SERS spectra, i.e., 3 different spectra were obtained for each sample. Finally, the background was removed using smoothing algorithms and the recorded spectra were normalized to an intensity value of 0 to 1. The range of the recorded spectra is 100–4278 cm^{-1}.

4.3 Mineral dataset

Mineral substance identification is another popular application of Raman spectroscopy. Among the various mineral databases, we selected the RRUFF database [25]. The RRUFF project aims at curating the most comprehensive set of high-quality spectral data from well-characterized minerals, comprising Raman spectra, X-ray diffraction, and information from chemistry. The database contains Raman spectra of various minerals at different configurations, i.e., varied wavelengths and orientations, accompanied by various kinds of processing. Furthermore, the Raman spectra computed from the different materials are hardly consistent, e.g., the ranges of Raman shifts are also different. In order to alleviate such irregularities, we have only considered spectra computed at 432 *nm*, which is the majority. Moreover, we have only collected the raw spectra, i.e., which were not processed anyway. Since the recorded spectra have different ranges of Raman shift, we have cropped the region that is common in all the spectra and used cubic spline to interpolate the locations if necessary. The resultant spectra cover the range 280–4237 cm^{-1}, and then, min-max normalization was performed. Finally, we take the mineral classes with at least 10 samples, which reduces the database to twenty mineral classes, and use them to evaluate the proposed model.

4.4 Bacteria dataset

We collected the bacteria-ID dataset from [12], where the potential of Raman spectroscopy in label-free bacteria detection was investigated. This dataset consists of 30 bacterial and yeast isolates, including multiple isolates of Gram-negative and Gram-positive bacteria. The dataset is organized into a reference training dataset, reference fine-tuning set, and test set. The fine-tuning dataset is used to account for the changes in measurement caused by optical system efficiency degradation. The training dataset contains 2000 spectra for each of the 30 isolates, whereas the fine-tuning and test set contains 100 spectra for each isolate. The isolates were cultured on blood agar plates sealed with Parafilm and stored at 4 $^{\circ}C$. The Raman spectra were generated using Horiba LabRAM HR Evolution Raman microscope, with 633 *nm* illumination at 13.17 *mW* along with a 300 *l/mm* grating. The spectra were computed at 1.2 cm^{-1} dispersion to simultaneously maximize signal strength and minimize background signal. The recorded spectra were normalized to the intensity of $0-1$, covering the spectral range between $381.98-1792.4$ cm^{-1}.

5 Experimental setup

The experiments have been conducted in a server computer with Intel Xeon @2.2GHz CPU, 24 GB RAM, and NVIDIA TESLA P100 (16 GB) GPU. We implemented the RamanNet architecture using Tensorflow [26]. The codes are available in the following GitHub repository.

https://github.com/nibtehaz/RamanNet

In the following subsections, we briefly explain the experimental protocols and evaluation procedures adopted for the different tasks. We have tried to mimic the corresponding baseline papers to ensure a fair comparison. As mentioned earlier, we were only compelled to follow a different evaluation scheme for the Mineral and Melanoma dataset, and thus we reproduced that baseline model's output following our exact protocol.

5.1 Covid dataset

We followed the same evaluation method as presented in [24]. In this work, the authors conducted a "blind" validation. They randomly divided the whole dataset into training (70%) and hold-out test set (30%). In order to further assess the independence of the data over model performance, this process was repeated 50 times and the average values of the metrics were recorded. We followed the same protocol to evaluate RamanNet. In order to avoid overfitting, we used 10% data from the training set as validation data, but the hold-out test data were left completely independent from the training process and were only used for evaluation. The models were trained for 1000 epochs.

5.2 Melanoma dataset

In the original work, Erzina et al. [13] performed a 75%:25% train-validation split. Therefore, we attempted to

follow a similar splitting criterion. Since the splitting information was not provided, we opted to perform a fourfold cross-validation instead, as it would also split the data in the same ratio. Additionally, in order to keep the comparisons fair, i.e., RamanNet has not benefitted by any convenient splitting that occurred randomly by chance, we have presented the results for all 4 folds. The authors demonstrated 100% accuracy using 3 different spectra (AuMs functionalized by ADT-NH_2, ADT-COOH, or ADT-$(COOH)_2$, respectively) together. However, to make the task difficult, we experimented with using only one type of spectra as input. To ensure a level-playing field, we reproduced the model as described in [13] and evaluated the model with one particular spectrum as input at a time.

5.3 Mineral dataset

Jichao et al. [9] used the RRUFF database for the mineral classification task. However, they employed the leave-one-out cross-validation scheme, which is computationally too expensive. In order to reduce the computational requirements, we thus opted for fivefold cross-validation instead. In order to compare RamanNet with the model presented by Jichao et al. [9], we implemented their proposed model and used it in our analysis.

5.4 Bacteria dataset

In order to assess RamanNet on the Bacteria-ID dataset, we followed the same training and evaluation procedure as presented in [12]. Similar to their approach, we first pre-trained the model using the reference training dataset, through a fivefold cross-validation scheme. The five models obtained in this process were then fine-tuned on the fine-tuning dataset, which was spilt into 90% training and 10% validation set. The model with the highest accuracy on this validation set was considered and evaluated on the independent test dataset. The models were trained for 100 and 250 epochs, respectively, on the reference training and fine-tuning set.

6 Results

6.1 RamanNet consistently outperforms existing models

6.1.1 COVID-19 dataset

In [24], a support vector machine (SVM) model was developed to distinguish the different categories, namely healthy, suspected, and COVID-19 patients. Instead of working with the entire spectrum, wave points with significant differences in the analysis of variance (ANOVA) test was selected. Thus, a statistically sound feature reduction was performed and the reduced feature set was used as the input to the SVM model. On the contrary, RamanNet takes the entire spectrum as input and adaptively finds the significant region therein.

The average performance over 50 random trials for the different tasks is presented in Table 2, and here we present the accuracy, sensitivity, and specificity values. Among the 3 tasks, differentiating between suspected and healthy subjects seems to be the most challenging one, as is evident from the inferior performance of SVM (all metrics $\leq$ 70%). RamanNet, on the other hand, performed comparatively better in this task. Notably, RamanNet improved accuracy and specificity by 13% and 21%, respectively. RamanNet also achieved an improved sensitivity score.

On the other two tasks, the SVM model performed comparatively (than its own performance in the first task). However, our proposed RamanNet consistently outperformed SVM in all these two tasks as well. Most promisingly, RamanNet improved sensitivity greatly in both of the tasks. For this problem, sensitivity is crucial as we need to correctly detect Covid-19 patients. This improvement in sensitivity did not come at any cost of specificity, rather the specificity has also been improved compared to the SVM model.

6.1.2 Melanoma dataset

As described previously, we perform a fourfold cross-validation on the melanoma dataset. Following the original evaluation as performed by Erzina et al., [13], we consider all three different types of spectra simultaneously as input. In addition, we perform a difficult version of the problem by taking only one type of spectra as input at a time. The results are presented in Table 3.

Table 2 Results on COVID-19 Dataset

Method	Accuracy	Sensitivity	Specificity
COVID-19 versus Suspected			
SVM	87 ± 5	89 ± 8	86 ± 9
RamanNet	93 ± 3	97 ± 4	90 ± 6
COVID-19 versus Healthy			
SVM	91 ± 4	89 ± 7	93 ± 6
RamanNet	95	95 ± 4	96 ± 3
Suspected versus Healthy			
SVM	69 ± 5	70 ± 9	66 ± 9
RamanNet	82 ± 6	77 ± 15	87 ± 11

Table 3 Results on Melanoma Dataset

	$-NH_2$		$-(COOH)_2$		–COOH		All		#Parameters	
Fold	Ours	CNN	Ours	CNN	Ours	CNN	Ours	CNN	Ours	CNN
1	**100**	97.42	100	100	**99.35**	94.19	100	100	1.3 M	25.7 M
2	**99.35**	96.13	**99.35**	98.71	**98.71**	96.12	**100**	98.71		
3	**100**	95.45	**100**	98.05	**99.35**	87.01	100	100		
4	**100**	97.40	**100**	98.70	**96.75**	96.10	**100**	99.35		

The bold numbers indicate the best performance achieved in the individual experimental settings

From the results, it is evident that although the CNN model manages to achieve perfect 100% accuracy (in 2 folds out of 4) when given 3 spectra as input, the accuracy falls when a single spectrum is given as input. This drop in performance can be explained by the loss of information when working with a single spectrum. For different functionalizations of AuMs, the sample is observed from a different point of view and different information is obtained. Thus, when working with a reduced number of spectra, insightful information is likely to get lost and that negatively affects performance. Although for the $-(COOH)_2$ as input the accuracy of the CNN model stays above 98%, it falls below 97% for the other two input spectra CNN model.

RamanNet on the other hand seems to consistently outperform the CNN model for both when all the 3 spectra are considered together or separately. RamanNet not only consistently achieved 100% accuracy with all the 3 spectra as input, but also with $-NH_2$ and $-(COOH)_2$ individual spectra as input separately. Only when –COOH spectra were used as input, the performance was not up to the mark, but still, the performance was superior to the CNN. All these improvements become more significant when we compare the number of parameters of the two models. The CNN model consists of 25.7 M parameters whereas RamanNet has only 1.3 M parameters ($\sim 5\%$). Thus, RamanNet is not only more accurate, but it is also computationally efficient at the same time.

6.1.3 Mineral dataset

As described in the previous sections, the Mineral dataset consists of 20 mineral classes, and in order to compare with the model proposed by [9], we implement their model and perform a fivefold cross-validation. The results are presented in Table 4.

Here, we have presented the Top 1, Top 3, and Top 5 accuracy metrics, which are popularly used for multiclass classification problems. In the Top X accuracy score, we check whether the top X predictions of the model match with the ground truth. It is evident that RamanNet consistently outperforms the previous state-of-the-art CNN model. For Top 1 accuracy, the improvement is more prominent, nevertheless, RamanNet performs better in regards to the other metrics as well. Another point worth mentioning is that even in this case the CNN model is heavier comparatively (6.6 M parameters vs. 1.3 M parameters for RamanNet).

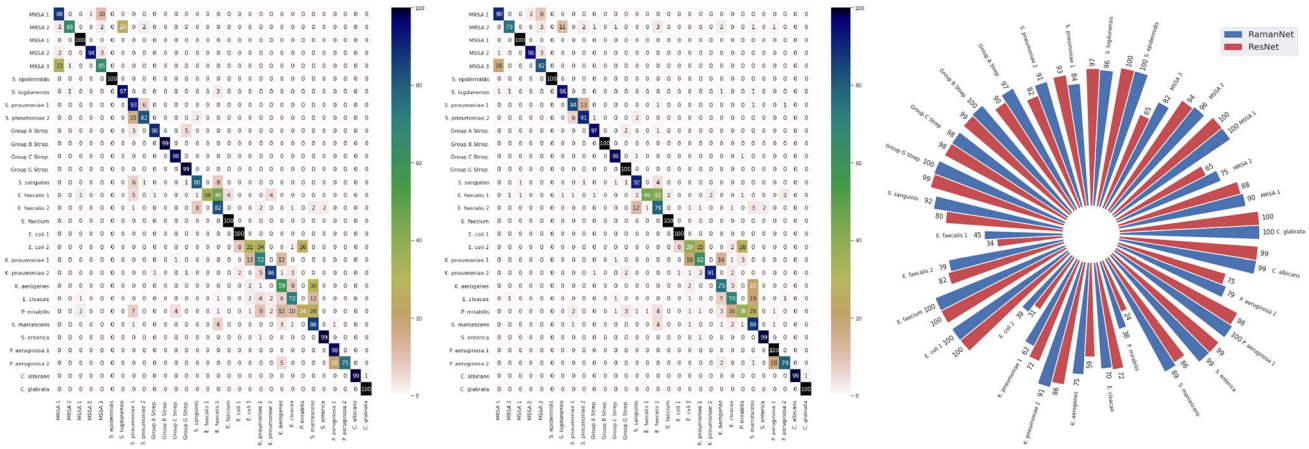
6.1.4 Bacteria dataset

In the original work [12], the authors used a 25-layer deep residual convolutional neural network to classify the bacteria isolate Raman spectrum to one of the 30 classes. The proposed model achieved an average accuracy of 82.2%, and the resulting confusion matrix is presented in Fig. 4a. The authors also experimented with common and popular classifiers like logistic regression (LR) and support vector machine (SVM) as baselines, but those models could only reach 75.7% and 74.9% accuracy, respectively.

Table 4 Results of Mineral dataset

	Top 1 accuracy		Top 3 accuracy		Top 5 accuracy		#Parameters	
Fold	RamanNet	CNN	RamanNet	CNN	RamanNet	CNN	RamanNet	CNN
1	**87.5**	81.25	**95.83**	87.5	**100**	93.75	1.3 M	6.6 M
2	**93.75**	85.42	97.92	97.92	**100**	97.92		
3	**89.58**	79.17	**93.75**	89.58	**93.75**	91.67		
4	**97.92**	85.42	**100**	91.67	**100**	93.75		
5	**85.12**	63.83	**93.62**	78.72	**95.74**	85.10		

The bold numbers indicate the best performance achieved in the individual experimental settings



(a) 25 Layer ResNet (b) RamanNet (c) Comparison

Fig. 4 Confusion matrix for Bacteria Dataset using 25-layer ResNet **a**, RamanNet **b** and their comparison **c**

RamanNet on the other hand manages to achieve an average accuracy of 85.5% on this dataset, despite only having 3 hidden layers. As shown in the confusion matrix (Fig. 4b), the number of misclassifications has reduced. Although erroneous predictions still exist, most Gram-positive and Gram-negative bacteria have been misclassified as Gram-positive and Gram-negative bacteria, respectively, and the errors are also mostly confined within the same genus, as analyzed in [12]. Although this 3.3% improvement may seem minor, this is achieved using a much shallower network (3 layers vs 25 layers). Furthermore, the improvements become more apparent when we compare the individual classifications next to each other. As presented in Fig. 4c, RamanNet achieves either equal or better accuracy for 25 out of 30 bacteria isolate classes. For the other two isolate classes, ResNet is only better with a small margin. On the contrary, in the cases where RamanNet performed better, it surpassed ResNet with a higher margin (e.g., Methicillin sensitive *Staphylococcus aureus* (MSSA 3), *Proteus mirabilis*, *Klebsiella aerogenes* etc.).

6.2 Triplet loss improves dimensionality reduction

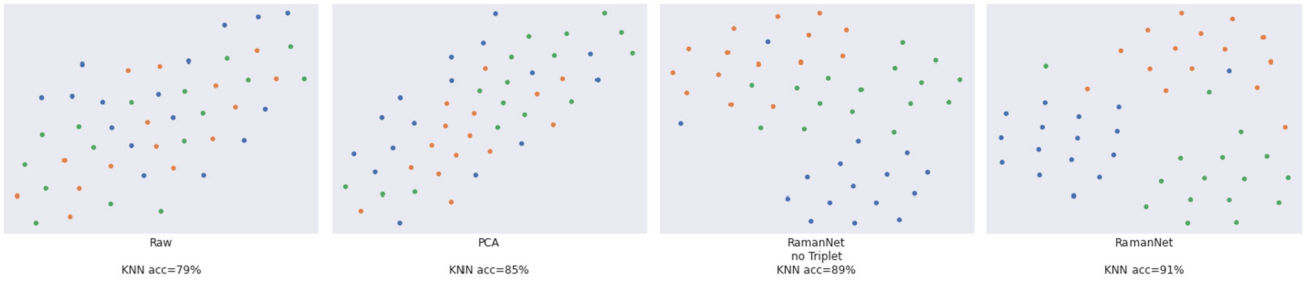
The majority of Raman spectra analysis works have been based on classical machine learning methods. Therefore, it has always been crucial to reduce the feature space. In this regard, principal component analysis (PCA) has been the most prominent method for feature reduction, to the extent that 34 out of the recent 52 papers used PCA [6]. Therefore, in the Raman spectra analysis community, feature selection and/or reduction is almost equally important as accurate classification.

Therefore, in order to perform the task of dimensionality reduction of Raman spectra, we have put focus on embedding generation capability of RamanNet. In addition to calibrating the embeddings learned by RamanNet from the class labels through backpropagated cross-entropy loss, we also include triplet loss in the embedding layer. This allows us to simultaneously minimize intraclass distance while maximizing interclass distance.

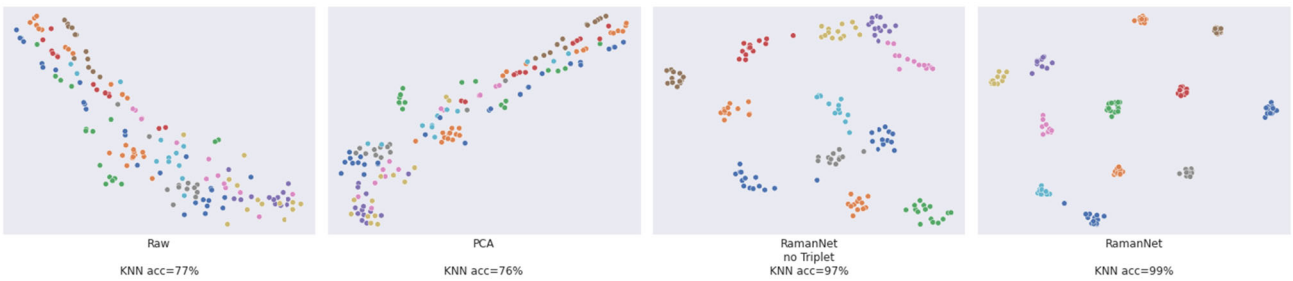
In order to assess the quality of the embeddings generated by RamanNet, we compare the RamanNet embeddings with PCA and the original raw spectrum. We also train a version of RamanNet without the triplet loss, to analyze the contribution of triplet loss. For qualitative analysis of the class separation obtained from such feature reduction, we plot 2-dimensional T-distributed Stochastic neighbor embedding (t-SNE) plots [27]. From Fig. 5, it can be observed that RamanNet embeddings are significantly superior to PCA or the original spectrum. Furthermore, RamanNet trained with triplet loss produces better embedding than training the model without this loss.

Since t-SNE is an approximate low-dimensional representation, we train simple KNN models with 15 neighbors and perform classification, as a mean of quantitative evaluation. Even in this case, it is evident that RamanNet trained with triplet loss is capable of the most desired feature representation. The results are presented in Table 5.

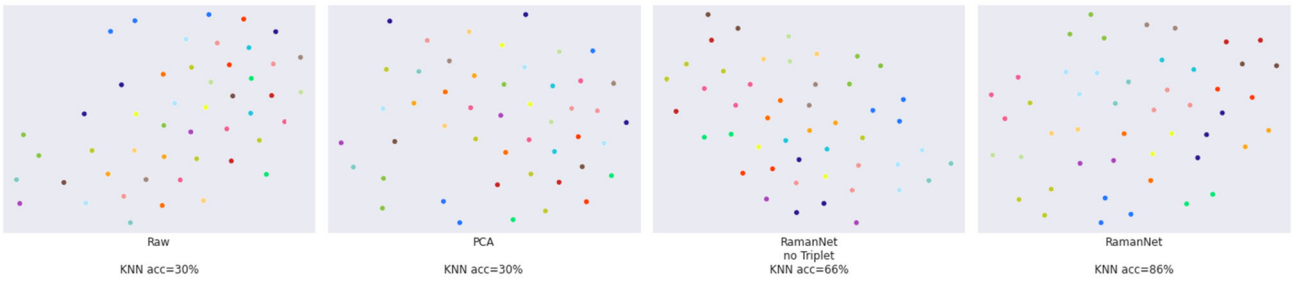
Fisher discriminant ratio (FDR) [28] is another measure of class separability. FDR provides a score of the features based on the centroids and spreads of the classes. For a dataset with C classes, each class i having n_i samples, suppose the mean and standard deviation values of a feature x_r are μ_i and σ_i, respectively, for class i. If the global mean and standard deviation of feature x_r are μ and σ, respectively, then the FDR value for that feature is defined as:



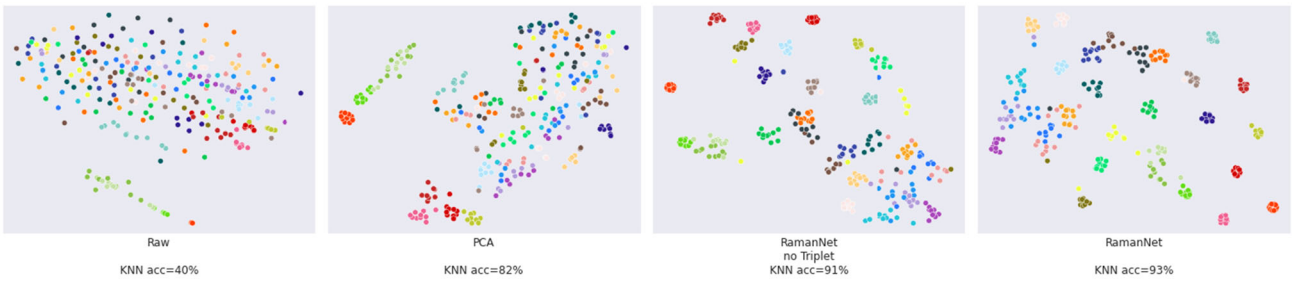
(a) Covid Dataset



(b) Melanoma Dataset



(c) Mineral Dataset



(d) Bacteria Dataset

Fig. 5 t-SNE embeddings of the Raman spectra of different datasets for different feature representations. Here, 2-dimensional t-SNE embeddings have been computed from the original raw spectrum, PCA features, RamanNet embeddings without and with triplet loss, respectively. In addition, we also report the accuracy of a simple KNN classifier for the individual feature representations

$$FDR_r = \frac{\sum_{i=1}^{C} n_i(\mu_i - \mu)^2}{\sum_{i=1}^{C} n_i \sigma_i^2} \tag{5}$$

A higher value of FDR indicates better class separability, whereas a lower value means that there exist overlaps

Table 5 Simple KNN model accuracy for various feature representations

		Dataset			
		Covid	Melanoma	Mineral	Bacteria
Feature representation	Raw	79%	77%	30%	40%
	PCA	85%	76%	30%	82%
	Without triplet loss	89%	97%	66%	91%
	RamanNet	91%	99%	86%	93%

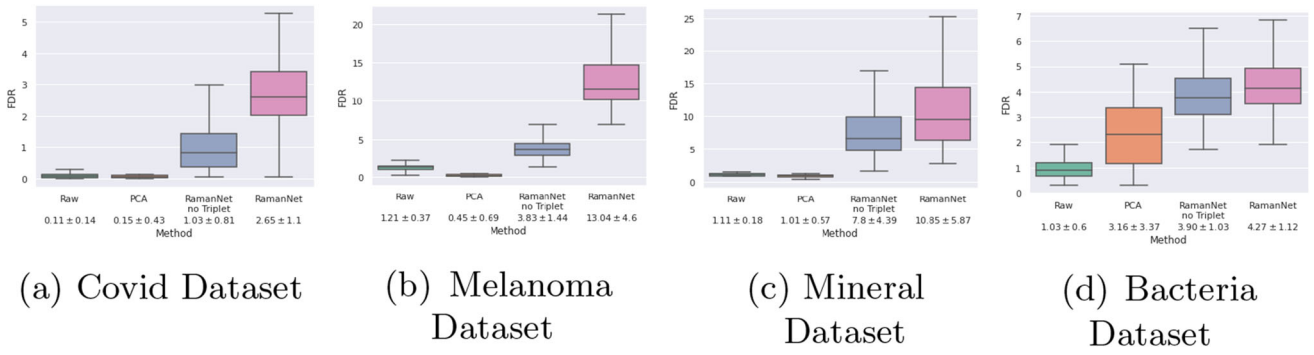


Fig. 6 Fisher discriminant ratio (FDR) scores for different feature representations

between the classes with respect to that particular feature. In order to assess the learned features quantitatively, therefore we compute the FDR values of the learned features and compare the scores with PCA and raw spectra. The comparisons are presented in Fig. 6. It is evident that RamanNet trained with triplet loss generates features with high FDR scores consistently. In the Mineral and Bacteria dataset, the improvement may appear less, this is because when the number of classes increases the notions of inter and intraclass distances gets a bit relaxed.

6.3 Model interpretation

Interpretability has been one of the focuses of deep learning research in recent years [29]. Deep learning models are competent function approximators and given a sufficient amount of data, they are capable of modeling almost any complex functions. With this potential, also comes the concern of what the model is actually learning from the data. The model can learn actual significant information and perform prediction accordingly, or it can merely learn from the noises and get confused by various confounding factors instead. Therefore, it is imperative to investigate the model's interpretability and examine what the model is learning from.

Compared to applications of deep learning in other domains, model interpretability is crucial in healthcare applications [32]. In general fields, model interpretability contributes to our understanding of how and what the deep models learn along with discovering potential approaches to make them more robust, accurate and free from biases. On the contrary, for medical applications, any machine learning model should be interpretable for the reasons of transparency, accountability, and regulatory compliance. Since such models decide life impacting decisions, the interpretation of those decisions is imperative.

For convolutional neural networks, we can use various methods like saliency maps [30] or score-CAM [31] methods for model interpretation. However, for multilayer perceptrons, it is non-trivial to do so. The various visualization methods have been designed based on CNNs and they cannot be directly translated to MLPs.

SHAP (SHapley Additive exPlanations) [44] is a game-theoretic approach to explain the output of machine learning models. The biggest advantage of using SHAP is that it is model agnostic, thus it can be used to analyze any machine learning model. Therefore, we can use SHAP to interpret the RamanNet model. In order to do so, we collected the features extracted from different sliding windows of RamanNet and trained the identical top layers using Scikit-Learn MLP implementation, for compatibility reasons with the official SHAP release [45].

We then computed the SHAP scores of all the $-NH_2$ samples in the melanoma dataset. We choose the melanoma dataset for this experiment because model interpretability is particularly crucial for disease diagnosis and information on significant biological properties related to melanoma was available. The results are presented as a violin plot in Fig. 7.

From the violin plot, it is evident that certain regions of the spectrum contribute most to the prediction. Moreover,

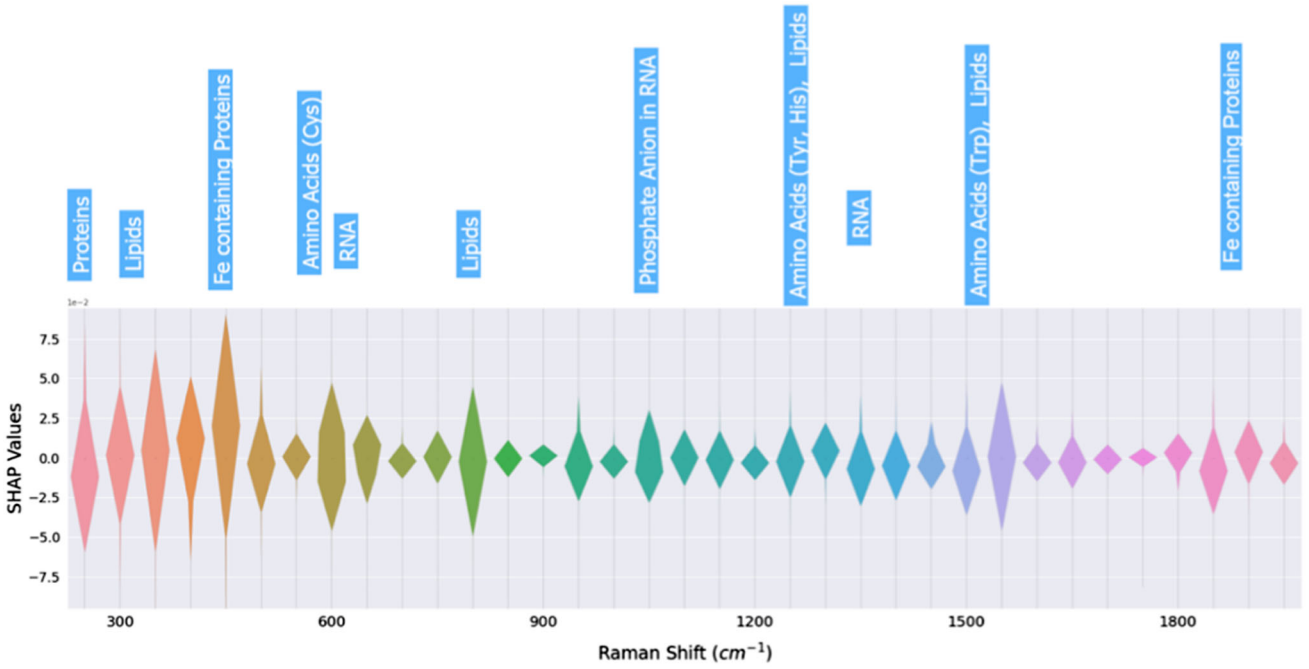


Fig. 7 SHAP values for RamanNet

those regions correspond to the actual significant properties of the sample. For example, the region in 400-500 cm^{-1}, corresponding to Fe-containing proteins, contributes the most to the prediction. After this region the lipids (300–400 cm^{-1}) contribute the most, and so on. The unlabeled regions in the figure correspond to less attributed regions, and it is apparent that the model was also aware enough to put less focus on them. All these findings are consistent with the one presented in [13]. Therefore, we can expect that RamanNet is learning significant information from the data and ignoring the noises instead of falling into a conundrum with confounding factors.

6.4 Computational complexity

In order to deploy a deep learning model for practical applications, computational complexity is an important concern. This criterion becomes more crucial when dealing with edge devices, i.e., incorporating a classification or processing model into Raman spectrum acquisition devices directly. In previous sections, it has been shown that the proposed RamanNet model is quite lightweight in terms of the number of parameters. In this section, the computational complexity of RamanNet is analyzed.

For deep learning models, the notion of computational complexity is different from classical algorithms. Notations such as, 'Big O' are less relevant as GPUs are used to parallelize the computation. In addition, the differences in GPU architectures and programming frameworks make it more complicated to compare the computational efficiency of two deep learning models.

FLOP or FLoating point OPeration is a basic unit of computation, which may represent an addition or multiplication. FLoating point OPerations (FLOPs) correspond to the total additions and multiplications involved in a computation, e.g., a single pass of a deep neural network for our purpose. Since FLOPs depend solely on the model architecture and are not influenced by the computational framework, it has become a popular measure of computational complexity [33]. FLOPs have a direct relationship with the computational complexity of the deep learning model. Regardless of hardware architecture or programming framework, the more FLOPs a model has the more operations it requires, i.e., the more time is consumed during training and inference.

In this work, RamanNet has been compared against 4 state-of-the-art methods. 3 of these methods are based on convolutional neural networks and the 4^{th} method is support vector machine (SVM). It is non-trivial to directly compare the computational complexity of a deep learning

Table 6 Comparison of computational complexity in terms of FLOPs

Model	ResNet	CNN-1	CNN-2	RamanNet
Task / Dataset	Bacteria	Melanoma	Mineral	4 tasks
Ref	[12]	[13]	[9]	This work
FLOPs	380 M	27.7 M	44.4 M	1.35 M

model with SVM for a multitude of reasons involving differences in programming frameworks and hardware platforms. Moreover, SVM is only competitive on small datasets free from outliers [43]. As the amount of data increases, deep learning models significantly outperform traditional machine learning models, which has led to their increased popularity and continuous development. Thus, we limit our analysis to the deep learning models. We computed the FLOPs of the state-of-the-art models for an input Raman spectrum of length 1000, which are presented in Table 6.

It can be observed that RamanNet only performs a fraction of computation compared to the other state-of-the-art models. Whereas the CNN models require $27-380$ million floating point operations to analyze a Raman spectrum of length 1000, RamanNet only needs to perform 1.35 million operations. This dramatic reduction in computational complexity makes the RamanNet model highly suitable for integration in Raman spectra acquisition devices. Therefore, not only RamanNet has a smaller number of parameters, but it is also efficient in using those parameters. The thoughtful use of MLP with sparse connectivity has indeed contributed to this computational efficiency.

6.5 Ablation study and hyperparameter tuning

In this section, we briefly analyze the effect of the different choices of various hyperparameters in RamanNet. The results for this section are computed on the 5th fold of the Mineral dataset, where the performance was comparatively worse. Thus, this enables us to understand the contributions of different hyperparameters better. The results are summarized in Fig. 8.

6.5.1 Selection of window length

The length of the window controls the degree of information processed by the network at the input level. Increasing the window length results in additional context whereas reducing it may enable the model to focus more on individual ranges of the spectrum better. Therefore, an optimal selection of window length is necessary for processing the spectra properly. From our experiments (Fig. 8A), increasing the window length, w from 10 to 50 gradually increases the accuracy as more contextual information is received. However, increasing w further worsens the performance with an increase of w the number of model parameters also increases, which may either lead to overfitting or sub-optimal training. Therefore, the value of w was selected as 50. It should be noted that the window step size, dw was selected as the half of w, following the standard in signal processing [42].

6.5.2 Remarks on window overlap

In signal processing, the standard approach is to process signals using overlapping windows [42], where after analyzing a window of length w, the next window is computed after a step size of $dw = \frac{w}{2}$. Therefore, this same protocol

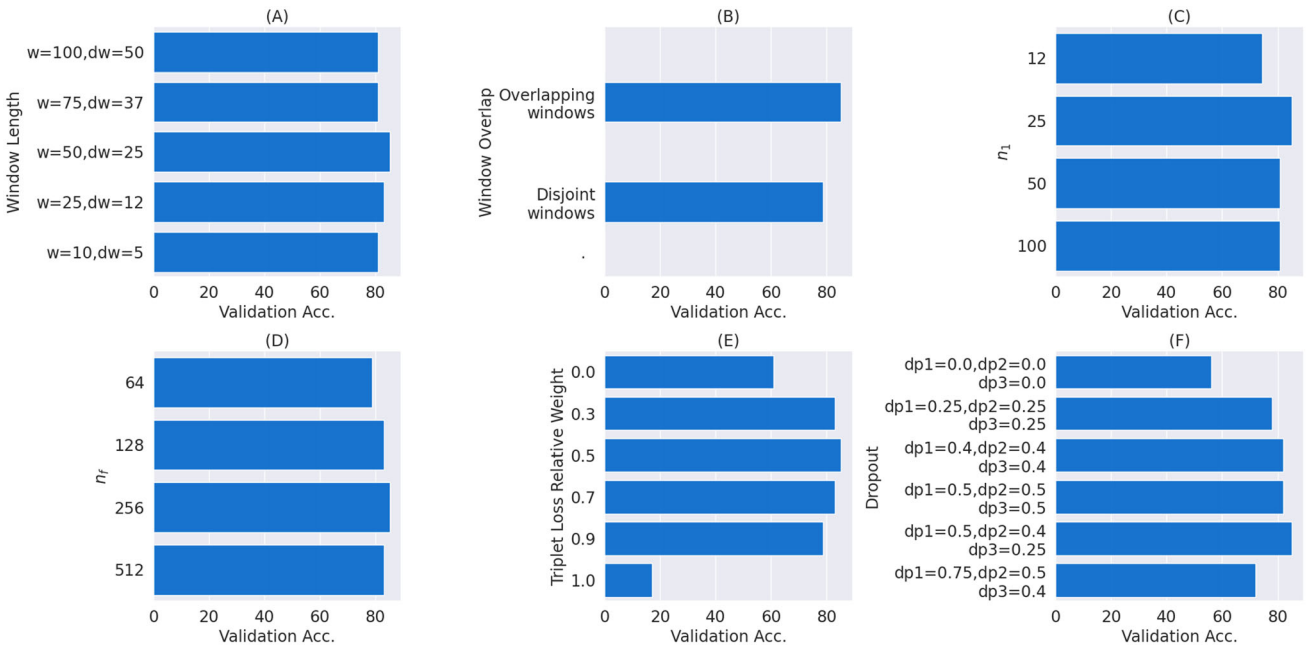


Fig. 8 Ablation Study and Hyperparameter Tuning

was followed in RamanNet. However, overlapping windows result in additional parameters in the model. So an experiment was conducted to observe the importance of this overlap at the input stage. From the experimental result (Fig. 8B), it is evident that removing the overlap in input significantly worsens the performance. This is likely due to the fact that the overlapping mode of windowing allows a portion of the spectra to be analyzed from two different contexts, i.e., the portion preceding it and the portion succeeding it.

6.5.3 Selection of number of neurons

In RamanNet, there are different numbers of neurons at different levels. For example, n_1 neurons at the input level and n_f neurons at the feature embedding level. In the initial stage of the development of RamanNet, different values of n_1 and n_f were experimented with. In both cases, it was observed that as the number of neurons is increased the performance starts improving, i.e., the expressive power of the model increases. However, after a certain point, the performance starts to fall, i.e., the model starts to overfit or inadequately fit the increasing number of neurons. Therefore, from experimental results 25 and 256 were selected from n_1 (Fig. 8C) and n_f (Fig. 8D), respectively.

6.5.4 Triplet loss weight

In order to overcome the noisy nature of Raman spectra, which makes them difficult to distinguish, triplet loss was adopted in the hidden layers. In previous sections, the efficacy of using triplet loss has been discussed through the analysis of class separability of generated embeddings. In this section, the relative weight of triplet loss compared to cross-entropy loss is investigated. Triplet loss and cross-entropy loss in our model share a symbiotic relationship, i.e., the first separates the classes in the feature space which enables the second one to classify them with ease. Therefore, removing the triplet loss completely (i.e., weight = 0) results in subpar performance. On the other extreme, removing cross-entropy loss (i.e., weight = 1) dramatically hampers the model performance as the model fails to learn classification. The weights in between appear to be helpful for the model. Interestingly, the weight of 0.5 for both losses resulted in the best accuracy score in our experiments and thus this weight was selected. If the weight is increased although the class separability improves the classification ability of the model gets reduced. On the other hand, reducing the weight generates less separable embeddings which makes the classification difficult. As a result, a proper balance between the two apparently performed the best in our experiments.

6.5.5 Dropouts

Dropout has been used in the RamanNet architecture as it is capable of reducing overfitting, which is prevalent in MLP networks. Several configurations of dropout were experimented with, which have been presented in Fig. 8f. Exclusion of dropout leads to overfitting and affects the performance negatively. On the other hand, integrating dropout layers improves performance. In our experiments, it was observed that gradually reducing dropout resulted in better models. Our rationale behind this is as we move from the input level to the embedding level, not only input noises are less prominent but also the feature maps contain useful information. As a result higher ratios of dropouts at earlier levels, which were necessary to suppress noises hamper information propagation at later layers. Thus, a smaller dropout probability was used in later levels. From the experiments, slowly reducing the dropout from 0.5 to 0.25 achieved superior performance to uniform dropout throughout the model.

7 Conclusion

Raman spectroscopy has slowly started to gain more attention with the advances in SERS technology. The gradual decrease in cost and complexity in computing the Raman spectrum is paving the way to large-scale Raman spectrum data collection for diverse tasks. Therefore, suitable machine learning methods are needed to analyze these large-scale Raman spectrum data. However, there has not yet been any model developed for the sole purpose of Raman spectroscopy analysis, motivated and designed based on the unique properties of the Raman spectrum. Recently, existing methods like CNN or SVM have shown success in Raman spectrum analysis, but in this work, we presented reasoning that such methods may not be adequately suitable for Raman spectra analysis.

In this work, we present RamanNet, a generalized neural network architecture for Raman spectrum analysis. We take intuitions from the nature of the Raman spectrum and design our model accordingly. We propose modifications to the convolutional network behaviors and emulate such operations using multi-layer perceptrons. This adjustment brings the best out of both worlds and it is reflected in the carefully designed experimental evaluation of the model on 4 public datasets. Not only that, the RamanNet outperforms all the state-of-the-art approaches in Raman spectroscopy analysis, it achieves it by adopting much less complexity. Moreover, RamanNet generates embeddings from spectrum which is much better than what is obtained from PCA, the de facto standard in Raman spectrum analysis. Furthermore, an interpretability study of RamanNet

particularly on a disease dataset and it was revealed that the model is capable of focusing on (biologically) meaningful information.

Nevertheless, some weaknesses or limitations of this work may be postulated. RamanNet uses MLP layers at the input level to analyze the spectrum which is not too different from CNN layers. The issue with such layers is that they can only learn a set of fixed filters and cannot account for variabilities in the input after a certain degree. Using self-attention in the input should be able to circumvent this limitation of the existing models, as it enables the model to learn adaptive weights based on inputs [41]. Additionally, the different segments of the spectrum are limitedly coupled during the computation of the model. This is true for both MLP and CNN-based models. Using transformer architecture has the potential to alleviate this limitation, since through query-key-value computation, different segments of the input are compared against the rest of the input [40]. All these can be incorporated in the successive developments of future iterations of RamanNet.

The future direction of this research can be manifold. Firstly, we wish to evaluate RamanNet on more large-scale datasets, as they become public. Pretraing deep learning models on a large possibly unlabeled dataset has become a popular paradigm in recent times, which can also be adapted for Raman spectrum analysis. Secondly, we have not performed any preprocessing or background removal of the spectra, we wish to investigate this further with a more dedicated study to infer the denoising capabilities of RamanNet. Last but not least, we also wish to experiment with multiple particle data to assess if RamanNet is capable of identifying, segmenting, and extracting the signatures of the different particles.

Acknowledgements This research is financially supported by Qatar National Research Foundation (QNRF), Grant number NPRP12S-0224-190144. The statements made herein are solely the responsibility of the authors.

Funding Open Access funding provided by the Qatar National Library.

Data availability All the datasets used in the experiments are publicly available.

Covid Dataset https://springernature.figshare.com/articles/dataset/Data_and_code_on_serum_Raman_spectroscopy_as_an_efficient_primary_screening_of_coronavirus_disease_in_2019_COVID-19_/12159924

Melanoma Dataset https://www.kaggle.com/datasets/andriitrelin/cells-raman-spectra

Mineral Dataset https://rruff.info

Bacteria Dataset https://github.com/csho33/bacteria-ID.

Code availability The codes are available in the following GitHub repository. https://github.com/nibtehaz/RamanNet

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Gardiner DJ (1989). Introduction to raman scattering. In Practical Raman Spectroscopy, pp 1-12. Springer
2. Raman CV and Krishnan KS (1928) A new type of secondary radiation. Nature 121(3048):501–502
3. Cornel J, Lindenberg C, Scholl J, and Mazzotti M (2012). Raman spec- troscopy. Industrial Crystallization Process Monitoring and Control, pp 93-103
4. Gordon G Hammes (2005). Spectroscopy for the biological sciences. John Wiley & Sons
5. Jones Robin R, Hooper David C, Zhang Liwu, Wolverson Daniel, Valev Ventsislav K (2019) Raman techniques: fundamentals and frontiers. Nanoscale Res Lett 14(1):1–34
6. Lussier F, Thibault V, Charron B, Wallace GQ, Masson J-F (2020) Deep learning and artificial intelligence methods for raman and surface-enhanced raman scattering. TrAC Trends in Anal Chem 124:115796
7. Wu X, Yiping Z, Zughaier SM (2021) Highly sensitive detection and differen- tiation of endotoxins derived from bacterial pathogens by surface-enhanced raman scattering. Biosensors 11(7):234
8. Braz A, Lopez-Lopez Maria, Garcia-Ruiz Carmen (2013) Raman spectroscopy for forensic analysis of inks in questioned documents. Forensic Sci Int 232(1–3):206–212
9. Liu J, Osadchy M, Ashton L, Foster M, Solomon CJ, Gibson SJ (2017) Deep convolutional neural networks for raman spectrum recognition: a unified solution. Analyst 142(21):4067–4074
10. Wu X, Chen J, Li X, Zhao Y, Zughaier SM (2014) Culture-free diag- nostics of pseudomonas aeruginosa infection by silver nanorod array based sers from clinical sputum samples. Nanomed Nanotechnol Biol Med 10(8):1863–1870
11. Shanmukh S, Jones L, Zhao Y-P, Driskell JD, Tripp RA, Dluhy RA (2008) Identification and clas- sification of respiratory syncytial virus (rsv) strains by surface-enhanced raman spectroscopy and multivariate statistical techniques. Anal Bioanal Chem 390(6):1551–1555
12. Ho CS, Jean N, Hogan CA, Blackmon L, Jeffrey SS, Holodniy M, Dionne J (2019) Rapid identification of pathogenic bacteria using

Raman spectroscopy and deep learning. Nature Commun 10(1):1–8
13. Erzina M, Trelin A, Guselnikova O, Dvorankova B, Strnadova K, Perminova A, Ulbrich P, Mares D, Jerabek V, Elashnikov R et al (2020) Precise cancer detection via the combination of functionalized sers surfaces and convolutional neural network with independent inputs. Sens Actuators, B Chem 308:127660
14. Yoo H-J (2015) Deep convolution neural networks in computer vision: a review. IEIE Trans Smart Process Comput 4(1):35–43
15. Kiranyaz S, Ince T, Abdeljaber O, Avci O, and Moncef G (2019). 1-d convolutional neural networks for signal processing applications. In ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp 8360-8364. IEEE
16. Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep Learning. MIT Press, 2016. http://www.deeplearningbook.org
17. Verleysen M and Francois D (2005). The curse of dimensionality in data mining and time series prediction. In International work-conference on artificial neural networks, pp 758-770. Springer
18. Schroff F, Kalenichenko D, and Philbin J (2015). Facenet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp 815-823
19. Maas AL, Hannun AY, Ng AY, et al. (2013) Rectifier nonlinearities improve neural network acoustic models. In Proc. icml, vol 30, pp 3. Citeseer
20. Ioffe S and Szegedy C (2015). Batch normalization: accelerating deep network training by reducing internal covariate shift. In: International conference on machine learning, pp 448-456. PMLR
21. Yan H, Yu M, Xia J, Zhu L, Zhang T, Zhu Z (2019) Tongue squamous cell carcinoma discrimination with raman spectroscopy and convolutional neural net- works. Vib Spectrosc 103:102938
22. Thrift WJ, Cabuslay A, Laird AB, Ranjbar S, Hochbaum AI, Ragan R (2019) Surface-enhanced raman scattering-based odor compass: lo- cating multiple chemical sources and pathogens. ACS sensors 4(9):2311–2319
23. Data and code on serum raman spectroscopy as an efficient primary screening of coronavirus disease in 2019 (covid-19). https://springernature.figshare.com/articles/dataset/ Data_and_-code_on_serum_Raman_spectroscopy_as_an_efficient_primary_screening_ of_coronavirus_disease_in_2019_COVID-19_/12159924/1, (Last accessed on July 2021)
24. Yin G, Li L, Lu S, Yin Y, Su Y, Zeng Y, Mei L, Maohua M, Hongyan Z, Lucia O et al (2021) An efficient primary screening of covid-19 by serum raman spectroscopy. J Raman Spectrosc 52(5):949–958
25. Lafuente B, Downs RT, Yang H, and Stone N. 1. the power of databases: The rruff project. In Highlights in mineralogical crystallography, pp 1-30. De Gruyter (O), 2015
26. Abadi M, Barham P, Chen J, Zhifeng Chen, Davis A, Dean J, Devin M, Ghemawat S, Irving G, Isard M, et al. (2016) Tensorflow: A system for large-scale machine learning. In 12th USENIX symposium on operating systems design and implementation (OSDI 16), pp 265-283
27. Van der Maaten L and Hinton G (2008). Visualizing data using t-sne. Journal of machine learning research, 9(11)
28. Rajoub B (2020). Chapter 2 - characterization of biomedical signals: Feature engineering and extraction. In Walid Zgallai, editor, Biomedical Signal Processing and Artificial Intelligence in Healthcare, Developments in Biomedical Engineering and Bioelectronics, pp 29-50. Aca- demic Press
29. Zhang Q and Zhu S-C (2018). Visual interpretability for deep learning: a survey. arXiv preprint arXiv:1802.00614
30. Ibtehaz N, Chowdhury MH, Khandakar A, Kiranyaz S, Rahman MS, Tahir A, Qiblawey Y, and Rahman T. Edith: Ecg biometrics aided by deep learning for reliable individual authentication. arXiv preprint arXiv:2102.08026, 2021
31. Rahman T, Khandakar A, Abdul KM, Islam KR, Islam Khandakar F, Rashid M, Tahir H, Tariqul IM, Saad K, Bin MZ et al (2020) Reliable tuberculosis detection using chest x-ray with deep learning, segmentation and visualization. IEEE Access 8:191586–191601
32. Amann J, Blasimme A, Vayena E et al (2020) Explainability for artificial intelligence in healthcare: a multidisciplinary perspective. BMC Med Inform Decis Mak 20:310
33. Howard, Andrew G., Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861
34. Voulodimos, A., Doulamis, N., Doulamis, A., & Protopapadakis, E. (2018). Deep learning for computer vision: A brief review. Computational intelligence and neuroscience
35. Lian, D., Yu, Z., Sun, X., & Gao, S. (2021). As-mlp: An axial shifted mlp architecture for vision. arXiv preprint arXiv:2107.08391
36. Ibtehaz, Nabil, and Mahmuda Naznin (2021). Determining Confused Brain Activity from EEG Sensor Signals. Proceedings of the 8th International Conference on Networking, Systems and Security
37. Hernández-Blanco, A., Herrera-Flores, B., Tomás, D., & Navarro-Colorado, B. (2019). A systematic review of deep learning approaches to educational data mining. Complexity
38. Otter Daniel W, Medina Julian R, Kalita Jugal K (2020) A survey of the usages of deep learning for natural language processing. IEEE Trans on Neural Networks Learning Syst 32(2):604–624
39. Miotto R, Wang F, Wang S, Jiang X, Dudley JT (2018) Deep learning for healthcare: review, opportunities and challenges. Brief Bioinform 19(6):1236–1246
40. Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin (2017). Attention is all you need. Advances in neural information processing systems 30
41. Niu Z, Zhong G, Hui Y (2021) A review on the attention mechanism of deep learning. Neurocomputing 452:48–62
42. Durak L, Arikan O (2003) Short-time Fourier transform: two fundamental properties and an optimal implementation. IEEE Trans Signal Process 51(5):1231–1242
43. Lai, Y (2019). A comparison of traditional machine learning and deep learning in image recognition. Journal of Physics: Conference Series. Vol. 1314. No. 1. IOP Publishing
44. Lundberg SM and Lee SI (2017). A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, Advances in Neural Information Processing Systems 30, pp 4765–4774. Curran Associates, Inc
45. Slundberg/shap (2021): A game theoretic approach to explain the output of any machine learning model. https://github.com/slundberg/shap. Accessed: November 7

Authors and Affiliations

Nabil Ibtehaz[1] · Muhammad E. H. Chowdhury[2] · Amith Khandakar[2] · Serkan Kiranyaz[2] · M. Sohel Rahman[3] · Susu M. Zughaier[4]

✉ Muhammad E. H. Chowdhury
mchowdhury@qu.edu.qa

✉ Susu M. Zughaier
szughaier@qu.edu.qa

Nabil Ibtehaz
nibtehaz@purdue.edu

Amith Khandakar
amitk@qu.edu.qa

Serkan Kiranyaz
mkiranyaz@qu.edu.qa

M. Sohel Rahman
msrahman@cse.buet.ac.bd

[1] Department of Computer Science, Purdue University, West Lafayette, IN 47907, USA

[2] Department of Electrical Engineering, Qatar University, 2713, Doha, Qatar

[3] Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Dhaka 1205, Bangladesh

[4] Department of Basic Medical Sciences, College of Medicine, QU Health, Qatar University, 2713, Doha, Qatar