Università degli studi di Milano-Bicocca

Advanced Machine Learning

Final Project

Deep Learning-Enabled Decoding of Raman Spectroscopy

Authors:
Nina Singlan - 867646- n.singlan@campus.unimib.it
Romain Michelucci - 867644- r.michelucci@campus.unimib.it

January 22, 2021

Abstract

Raman spectroscopy is commonly used in chemistry to provide a structural fingerprint by which molecules can be identified. Machine Learning (ML) methods are used to decode the resulting spectra. The purpose of this study is to identify patients affected by Amyotrophic Lateral Sclerosis (ALS) using ML methods. The main problem for an efficient classification is the data scarsity. Indeed, Raman spectra are subject to medical confidentiality. Regarding the limited dataset, a model pre-trained on a larger one, composed of bacterial spectra, is a good solution to obtain better performances. This report presents a Transfer Learning (TL) approach combined with data augmentation techniques.

1 Introduction

Raman spectroscopy allied with ML methods promises a new, rapid and non-invasive technique to diagnose patients. This ability has led researchers to highlight that Convolutional Neural Networks (CNNs) outperform other ML methods. However, CNNs are ”data-hungry”, they require a massive amount of data in order to extract the right features. Usually, a CNN can be well-trained on a few thousand samples.

Unfortunately, the dataset is composed of less than 600 spectra, which is not really sufficient to properly train a model. In addition, a problem lies in the repartition of the number of samples per patient and per group. Transfer Learning aims at improving the performance on a target domain by transferring the knowledge contained in different but related source domains.

In this work, the effort is focused on trying different TL experiments. The first is fine-tuning which consists of unfreezing a pre-trained models and re-training it on the new data. The second experiment is the most common incarnation of transfer learning : features extraction. After taking the layers from a pre-trained model, the layers are freezed and some new trainable layers are added on top of the frozen ones. These new layers will learn to turn the old features into predictions on a new dataset.

Finally some data augmentation methods will be applied. It is a well-known technique for improving the training of neural networks. The idea is to expand the number of training samples by adding some noise and small variations resulting in a more robust training. For spectral data, random offsets, random multiplications and Gaussian noises are commonly used.

2 Datasets

The dataset provided is composed of 591 spectra, belonging to 30 patients, divided into two classes : patients affected by ALS and healthy ones (CTRL). 393 spectra are provided for the ALS patients and 198 for the CTRL ones as it is shown in table 1. Thus, it is quite unbalanced. Furthermore, notice that the spectra of a same patient are correlated, so it is necessary to treat them together.

Thanks to Dario Bertazioli who provided the data, they are filtered and pre-processed. The only addition is the negative values removal from the spectra. In fact, these points are meaningless, having negative value would literally mean that energy is produced by scattering.

The bacteria-ID dataset, on which the models used for TL has been trained, consists of 30 bacterial and yeast isolates. 2000 spectra are associated for each of the 30 reference isolates.

3 The Methodological Approach

The first core idea of this project was the comprehension of the dataset and its limits.

At the beginning, 4 classical ML classifiers have been implemented :

- Decision Tree Classifer
- Logistic Regression
- SVM
- Random Forest Classifier

If the correlation of spectral data among the same patient is ignored, the models are overfitted caused by a data leakage. Therefore, to properly have an unbiased diagnostic method, there is a need to carefully split the training and test data. It is mandatory that spectra belonging to the same patients don't show up both in the training set and in the test set. To maximaly exploit the available training data, a Leave-One-Patient-Out Cross Validation and a 8 GroupKFold on patients have been implemented. It is worth mentionning that some feature preprocessing has also been done with Principal Component Analysis (PCA) which seems to increase the performance

Table 1: This table represents the repartition of patients and spectra belonging to the ALS and CTRL groups

Repartition of spectra and patients in dataset				
Groups	Patient ID	Samples range	Samples count	Number of samples per group
ALS	ALS01	1-60	60	393
	ALS02	61-78	18	
	ALS05	79-114	36	
	ALS07	115-150	36	
	ALS08	151 -194	44	
	ALS09	195-210	16	
	ALS10	211-225	15	
	ALS11	226-242	16*	
	ALS12	243-256	14	
	ALS13	257-281	25	
	ALS14	282-300	19	
	ALS15	301-314	14	
	ALS16	315-324	10	
	ALS17	325-334	10	
	ALS18	335-344	10	
	ALS20	345-354	10	
	ALS22	355-364	10	
	ALS23	365-374	10	
	ALS24	375-384	10	
	ALS25	385-394	10	
CTRL	CTRL01	1-33	33	198
	CTRL02	34-76	43	
	CTRL03	77-91	15	
	CTRL04	92-138	47	
	CTRL05	139-149	11	
	CTRL06	150-158	9	
	CTRL07	159-168	10	
	CTRL08	169-178	10	
	CTRL09	179-188	10	
	CTRL10	189-198	10	

* The 227th sample is missing. It has probably been removed while processing because it was ”too bad”.

of the models (except SVM because the kernelized machine is already doing a similar work).

After setting up these baselines, the second and central step was the discovery and the application of the methods used in the bacteria-ID paper. In this respect, some basic predictions have been firstly computed on the raw dataset using the three pre-trained models (CPKT extension):

- pretrained model
- finetuned model
- clinical pretrained model

There are saved parameters for the pre-trained CNN which is based on the ResNet architecture and dynamically modified to fit the input and output dimensions.

Then, the fine-tuning experiments step resulted from the observation that none of the model obtain good performances. It is due to the difference between the source and target domains of the training set.

Given the custom splitting method presented in the bacteria-ID paper the patients are divided into sub-groups of 5 patients each. The samples of the first three are used for the training set, the fourth for the validation set and the remaining for the test set. Each of the samples selected are then shuffled to make sure that no links may be found by the models. Some of which are selected for finetuning the models and the remaining for predicting on the finetuned models. However as we have a lack of data it is not possible to apply this method (some of the patients have only 9 samples) Therefore, a more suitable method has been developped. First, the data is divided into:

- a ”finetunable” dataset for finetuning the pretrained models. It is composed of 20 patients.
- a ”full test” set which is used for computing the performances of the finetuned models. It is composed of 10 patients.

The repartition among the patients have to be wisely selected due to data scarsity. Each of the groups has to be well represented into the 2 datasets. A raisonable ratio in terms of patient would be 12/8 for ALS (i.e. 8/2 for CTRL) or 14/6 for ALS (i.e. 6/4 for CTRL).

It appears that it increases in terms of accuracy performances and more stability arises (the standard deviation is lowered).

But, as shown after multiple iterations, the choice of the patient repartition (and implicitely the number of samples of one group over another) have great influences on the final result. A finetuned model which has been trained on a bigger amount of ALS samples will have a big problem to deal with CTRL when predicting. Therefore the results are biased. However the case when the finetuned model is trained on a bigger amount of CTRL never happens !

An other TL experiment is made, using the technique of "features extraction". That means the pre-trained models are used, not to predict classes, but to extract features from data.

To this purpose, each of the three provided models are cut at different levels :

- After the last block : in this case, just the Linear layer is removed. This kind of model extract 37 for each spectrum.
- Before the last block : here, all the last block is removed. That is resulting 74 features for each spectrum.
- Before the two last blocks : this is removing the entire two last blocks of the pre-trained model. It is producing 147 features for each spectrum.

From this technique, 9 features extractors are obtained, taking the data as input ang returning features as output. To obtain prediction from these, two different models are proposed :

- Classical ML model : Decision Tree Classifier
- Deep ML model described in table 2.

Table 2: This table describes the deep ML model created

Layer type	Input	Output
InputLayer	Number of features extracted	Number of features extracted
DenseLayer	Number of features extracted	128
DenseLayer	128	64
DenseLayer	64	32
DenseLayer	32	16
Dropout	16	16
DenseLayer	16	1

On classical models, this techniques increase the performances. On deep models, the same problem of data is raised. Indeed, when data are randomly "well" distribued, this method can achieve really good performances, but when data are not, very bad performances are noticied.

The repartition among patients is not the core problem. The number of spectral samples per patient influence drastically the final result. So, the only conclusion possible is to develop some techniques of data augmentation in order to have :

- 1. a better repartition of samples per group (ALS/CTRL)
- 2. a better repartition of samples per patient (70 ¿¿ 9 !)

This led us to the final step : the data augmentation techniques. 3 basic ones have been tested:

- Random Offset
- Random Mulitplication
- Gaussian noise

Although low the performances are increased as shown in the following section.

The data augmentation techniques previously described are also applied with features extraction. For runtime reasons, only one features extractors is used, which is the one obtained from the "Finetuned pre-trained model" with a cut before the last block. As expected, the dataset obtain is more balanced, which allow more constant results for models.

4 Results and Evaluation

The Results section is dedicated to presenting the actual results (i.e. measured and calculated quantities), not to discussing their meaning or interpretation. The results should be summarized using appropriate Tables and Figures (graphs or schematics). Every Figure and Table should have a legend that describes concisely what is contained or shown. Figure legends go below the figure, table legends above the table. Throughout the report, but especially in this section, pay attention to reporting numbers with an appropriate number of significant figures.

5 Discussion

The discussion section aims at interpreting the results in light of the project's objectives. The most important goal of this section is to interpret the results so that the reader is informed of the insight or answers that the results provide. This section should also present an evaluation of the particular approach taken by the group. For example: Based on the results, how could the experimental procedure be improved? What additional, future work may be warranted? What recommendations can be drawn?

6 Conclusions

Conclusions should summarize the central points made in the Discussion section, reinforcing for the reader the value and implications of the work. If the results were not definitive, specific future work that may be needed can be (briefly) described. The conclusions should never contain "surprises". Therefore, any conclusions should be based on observations and data already discussed. It is considered extremely bad form to introduce new data in the conclusions.

7 Data and code availability

All the data and the code are available on GitHub

References

The references section should contain complete citations following standard form. The references should be numbered and listed in the order they were cited in the body of the report. In the text of the report, a particular reference can be cited by using a numerical number in brackets as [**?**] that corresponds to its number in the reference list. LaTeXprovides several styles to format the references