Classifying rocks via 7810-dimensional spectroscopy

Miroslav Vitkov

January 27, 2023

Abstract

This work explores classification of rocks by their laser ablation spectrum in C++.

Multiclass classification over sparse input space is performed[1]. The task is made significantly easier by having up to 64 input vectors with the same label. The most promising algorithm is then tuned - kernel type, kernel parameters, data preprocessing. The business application accuracy is theoretically derived from the per sample accuracy.

A secondary goal is overview of modern free C++ machine learning libraries.

1 Introduction

Geologists recognise three categories of rocks: *sedimentary, metamorphic, and igneous*. Each containing tens of concrete types of rock. Measuring the type of a rock is currently being done by bulky equipment after rock sample collection. This text explores the machine learning side of designing a device, which is:

- hand-held
- accurate to less than 1 failure per year of constant usage
- afordable - an Android smartphone used for data processing.

One competitor of the device are the Nitron X-ray handhelds[10].

The theoretical conclusion is that the device is possible.

2 Related Work

Rocks from the Karkonosze Mountains have been classified via [350 nm−2500 nm] spectroscopy[6]. They acheaved a macro averaged accuracy of 76.2%. A standartised light source was used instead of laser ablation.

[1]https://github.com/MiroslavVitkov/rocks

A lidar-based survey[1] produced inconclusive results.

The Curiosity mars rover was equipped with a LIBS unit[9] as will be the upcoming ExoMars[8].

Interestingly, LIBS seems to be pretty loud and can in the future benefit from fusing the spectrogram with acoustic features[2].

3 Dataset

Laser-induced breakdown spectroscopy[4] involves turning some matter into plasma and observing it's radiant spectrum. That is measured by a spectrometer and discretized into 7810 buckets of central frequency from 180 nm upward in steps of 0.1 nm. The spectrometer has an apparent peak in sensitivity around 440 nm but important spectral lines are spread through the whole measurement range.

Key variables in LIBS are beam power, diameter; burn time(the spectrum changes with time during the burn). They are held constant and unknown in this study.

The 7810 measured values for each spectrum represent the radiance[7] of the plasma convoluted with the **unknown** impulse response of the measurement equipment.

The dataset consists of 2710 datapoints all classified within 6 labels(rock types) and 45 sublabels(measurement neighbourhoods). The reference number of the dataset is 190401. The excitation laser wavelength is 1064 nm.

The string labels are encoded in integers in alphabetical order:

azurite	0
brochantite	1
chalcopyrite	2
chrysocolla	3
malachite	4
tetrahedrite	5

The sub-labels are simply the consecutive number of the measured spot.

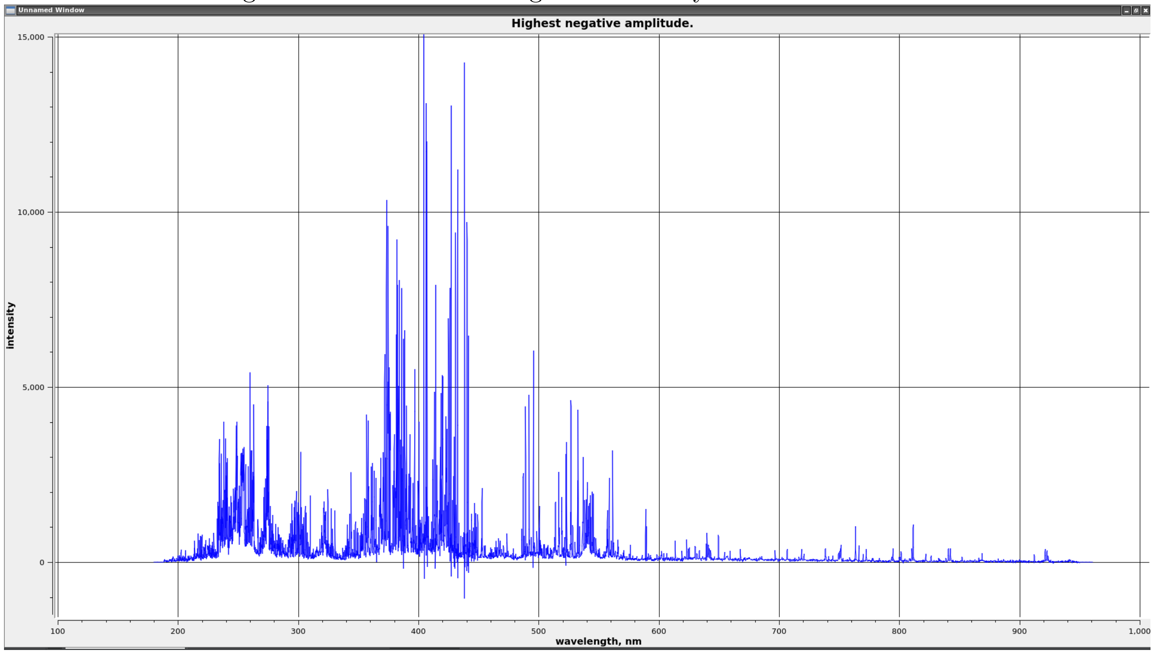
4 Noise

Boiling plasma is not theoretically expected to absorb light of any wavelength.
The dataset violates this assumption!

```
Dataset consists of 2710 files.
The global micro average intensity is 114.398.
The global count of negative values is 2442125, which is 0.115385 of all datapoints.
The mean of all negative values is -9.10135.
The most extreme negative value is -1042.48.
```

s

Figure 1: Most extreme negative intensity



- One model of the noise is harmonics of the exciting laser. This hypothesis fails due to it operating at 1064 nm, which is not a multiple of the observed spikes' frequency.
- Another model is Gaussian noise with $\mu = 440\,\text{nm}, \sigma = 40\,\text{nm}$.
- Another model is uniform noise, multiplied by the impulse response of the measurement equipment. Instrument IR can be recovered by averaging each frequency's mean value over all samples (i.e. stochastic multidimensional mean).

Noise filtering has not been implemented. The models are expected to cope with the noise on their own.

5 Dimensionality Reduction

The number of features (7810) is larger than the dataset size(2711). Thus better performance is expected from the classifiers after dimensionality reduction.

On the other hand, feature selection techniques should point to prominent spectral lines. Those could be valuable to domain experts.

5.1 Principal component analysis

Numerous algorithms have been developed to compute this simple yet powerful transformation. For example the Gram-Schmid ortogonalization can be parallelised over a GPU [3]. Here we use the classical NIPALS-PCA algorithm for simlicity.

5.2 Linear discriminant analysis

$$\frac{\mu_1 - \mu_2}{s^1 + s^2} \rightarrow max$$

Like *sklearn*'s LDA implementation, *OpenCV*'s results in at most (num classes - 1) dimensional transformed space. For the examined dataset *d=5*. The classification accuracy of the SVM dropped from 0.97 to 0.18.

5.3 t-SNE

For comparison, a t-distributed stochastic neighbour embedding should have been run.

6 Preprocessing

6.1 Normalization

$$\vec{y} = (\vec{x} - \vec{E})/\vec{\sigma}$$

, where all operations are performed element-wise. Normalization surprisingly worsened performance from 0.97 to 0.93.

6.2 Logarithm

Compressing the signal range worsened the accuracy from 0.97 to 0.77.

7 Models

Each model conceptually accepts a single spectrum vector of 7810 real numbers. Later in this text the sampling strategy and ensemble voting are discussed. Each algorithm is trained on a labeled dataset (split from the original dataset via simple holdout). And then evaluated on a similarly constructed test dataset, one spectrum (7810-dimensional vector) at a time.

The following algorithms were tried with default parameters, this is the micro averaged accuracy:

algorithm	accuracy	predict	train	acceleration
random chance	0.16	0 s	0 s	single thread
correlation	0.85	76 s	0 s	multi-thread
SVM	0.97	0 s	3 min 21 s	multi-thread
neural network	0.20	0 s	2 min 7 s	multi-thread
random forest	0.17	21 s	1 h 49 min	multi-thread

7.1 Random Chance

For sanity checking of the system, a random chance model was developed. It looks only at the distribution of training labels and produces a similar stochastic distribution at prediction time. An alternative would have been to always predict the most common class. That would have improved repeatability between runs, but the generated distribution would have been too different than that of a predictive model.

7.2 Correlation

With such a small dataset, correlation is feasible. It yields good results, but is not scalable.

7.3 SVM

A multiclass SVM is tested with a linear kernel because that is the only one dlib[5] supports.

7.4 Neural Network

A naive model with two hidden layers with relu activations performed extremely poorly. It is unclear if this is due to programming errors or due to wrong architecture of the network.

7.5 Random Forest

An OpenMP-based random forest classifier took extremely long to train and yielded poor results.

8 Measuring Accuracy

Accuracy, as well as the other summary statistics, in the case of multiclass classification is not uniquely defined. Those are the two approaches:

- Micro average - record a True Positive and Total statistics among all classes.
- Macro average - compute the accuracy for each class, then average those.

Our approach is to calculate a confusion matrix because of it's usefulness in gaining insight into the problem. Then reduce that to a signle number: micro averaged accuracy - so that models can be compared.

9 Sampling

Holdout sampling is performed thus: 66% of all datapoints across all labels are selected as a training set. The remaining 34% are assigned to a test set with no validation set assigned. The split is performed via a pseudorandom number generator because the application does not need to be cryptographically secure. No regard is payed to labels when splitting.

10 Tuning the model

The SVM model performed well and runs quickly at prediction time. This model was selected for parameter tuning. However, the used implementation allows the use only linear kernels for multiclass SVMs (why)? A grid search for the regularization coefficient c in the range [1e-5, 1e5] indicates said space is rather flat.

c	accuracy
1e-5	0.788652
1e-4	0.780142
1e-3	0.798582
1e-2	0.794326
1e-1	0.798582
1e0	0.802837
1e1	0.792908
1e2	0.801418
1e3	0.791489
1e4	0.807092
1e5	0.798582

Also notice the drop in accuracy compared to the results reported in chapter 'Model'. This is due to training on a smaller dataset, due to withholding the validation partition. Naturally, both the training and validation partitions sets are used to train the final classifier.

11 Implementation

11.1 Language

C++ was chosen as the language of the implementation.

11.2 Design

The program uses the **Command** design pattern to fit different working modes into one executable.

The classifier models follow the Open/Close OOP principle. They should have worked with either raw spectra or with preprocessed data, but that would have complicated the interface design. Thus preprocessing techniques were tested in hardcoded SVMandX models.

To separate interface from implementation **pImpl** is used. Not only do implementation details remain hidden, external includes also remain only in the .cpp file.

The **Factory** model::create() connects seamlessly with the command line parser module.

11.3 Libraries

- dlib - a small mathematical library, used for correlation, neural networks, SVM.
- Qwt - a native plotting library. It's main rival is the C++ bindings for pyplot.
- OpenCV - used for PCA, because dlib's methods only work for binary classification problems out of the box.

11.4 Data structures

Traditionally a dataframe is used to store and represent a dataset. To the contrary we used a map from labels to a set of observations. A function was provided for walking this structure. Overall this setup performs well except when the shape of the dataframe needs to change.

12 Voting

Majority voting with n=60 samples and accuracy a=0.9828 yields overall expected error rate of

```
$ find . -name *.csv | wc -l
2710
$ calc "2710 / 45"
~60.22222
>>> import math
>>> # Probability of majority voting to fail:
>>> # ( 1 - 0.9828 ) ^ 30 + ( 1 - 0.9828 ) ^ 31 + ... + ( 1 - 0.9828 ) ^ 60
>>> # Let's take only the largest term.
>>> math.pow(( 1 - 0.9828 ),30)
1.163733153901975e-53
```

```
>>> # Let's assume one test per second.
>>> # The Mean Time Between Failures is the invese of the frequency of failures.
>>> 1 / 1.163733153901975e-53
8.593035238766027e+52  # seconds
>>> # Let's divide this by the age of the universe.
>>> 8.593035238766027e+52 / ( 31557600 * 13.8e9 )
1.973165617645385e+35
```

13 Conclusion

13.1 Experimental

A large number of test points which are guaranteed to be of the same label is provided during real world operation. This results in even a simple model like an SVM solving the problem practically perfectly. A typical confusion matrix looks like this:

```
52  0  0  0  0  2
 0 63  0  0  0  0
 0  0 57  1  0  0
 0  0  0 64  0  0
 0  0  0  4 59  1
 2  0  0  0  0 62
```

13.2 Language

Customarily machine learning is done in Python. Those are our observations over the suitability of C++ for such tasks.

- Central repository - Python "comes with the batteries included". Somewhat mitigating this, this project uses a plethora of libraries known to be open-source, with permissive license, easy to use, efficient. The project thus can be used as a demo of C++ ML libraries.
- Type safety - due to the nature of matrix multiplication, the result is rarely in the same form as the input. Thus new types need to be created between any two operations, making pipelining more tedious. Overall little is gained by labelling lists of numbers which change their representation but preserve their meaning.
- Plotting - plotting libraries for C++ are scarce, one of the most popular candidates being the bindings to matplotlib.
- Dataframes - matrix calculations, consistency between test run results.
- GPU access - OS and hardware abstraction is almost impossible

References

[1] Leonardo Campos Inocencio et al. “Spectral Pattern Classification in Lidar Data for RockIdentification in Outcrops”. In: *The Scientific World Journal* (2014).

[2] S. Maurice et al. “A Microphone Supporting LIBS Investigation on Mars”. In: *LunarandPlanetaryScienceConference* 47 (2016).

[3] M. Andrecut. “Parallel GPU Implementation of Iterative PCA Algorithms”. In: *Journal of Computational Biology* 16 (Dec 2009).

[4] *Introduction to LIBS*. `https://www.nist.gov/pml/atomic-spectra-database`. (Visited on 07/19/2019).

[5] Davis E. King. “Dlib-ml: A Machine Learning Toolkit”. In: *Journal of Machine Learning Research* 10 (2009), pp. 1755–1758.

[6] Monika Mierczyk et al. “”Assessment of Imaging Spectroscopy for rock identification in the Karkonosze Mountains, Poland””. In: *Miscellanea Geographica* 20 (2016), pp. 34–40.

[7] Palmer and M. James. “The SI system and SI units for Radiometry and photometry”. In: *Information Processing and Management* (2012).

[8] Fernando Rull, Berit Ahlers, and Gregory Bazalgette Courreges-Lacoste. “Combined Raman spectrometer/laser-induced spectrometer for the next ESA mission to breakdown Mars”. In: *Spectrochimica Acta Part A Molecular and Biomolecular Spectroscopy* 68 (2008), pp. 1023–8.

[9] B. Salle et al. “Comparative study of different methodologies for quantitative rock analysis by Laser-Induced Breakdown Spectroscopy in a simulated Martian atmosphere”. In: *Spectrochimica Acta Part B-Atomic Spectroscopy* 61 (2006), pp. 301–313.

[10] *Thermo Scientific™ Niton™ portable X-ray fluorescence (XRF) analyzer.* `https://www.thermofisher.com/bg/en/home/industrial/spectroscopy-elemental-isotope-analysis/portable-analysis-material-id/portable-mining-exploration-solutions/portable-hard-rock-mining-analysis.html`. (Visited on 04/18/2020).