21st International Conference on Pattern Recognition (ICPR 2012)
November 11-15, 2012. Tsukuba, Japan

Finding Discriminative Features for Raman Spectroscopy

Michael Kemmler Joachim Denzler
Chair for Computer Vision, Friedrich Schiller University of Jena
{*michael.kemmler, joachim.denzler*}*@uni-jena.de*

Abstract

To identify microorganisms is of utmost importance in various applications such as medical science and pharmaceutical industry. The technique of Raman spectroscopy is particularly useful in this scenario, since it extracts a high-dimensional molecular fingerprint from samples at hand. Instead of using the complete spectrum, it is often sensible to concentrate on a small number of discriminative dimensions. Apart from providing important molecular insights, this can be beneficial in terms of speed and accuracy. This work studies several state-of-the-art machine learning techniques suitable for feature ranking, many of which have not been used before in the context of Raman spectra classification. Experiments on three different bacteria classification problems show that boosting-based methods and zero-norm support vector machines are especially suited for this challenging task.

1. Introduction

Given the large biodiversity of microbiological species, the identification of microorganisms is a challenging task. Raman spectroscopy is an optical technique for measuring molecular vibrations that has recently received much interest in this field. Combined with powerful classifiers, it has been shown to enable an accurate and fast analysis of microorganisms [12]. Raman spectra (see Figure 1) usually contain many hundreds or a few thousand dimensions. Finding relevant spectral features is therefore desired for several reasons. First, concentrating on a few features can increase speed and accuracy. Second, one spectrum essentially is a superposition of responses from molecular bonds, each of which can be traced back to certain frequencies. Finding discriminative features can therefore provide information to the biologist about chemical compounds, which are relevant for a given task.

The search for relevant features can be realized via

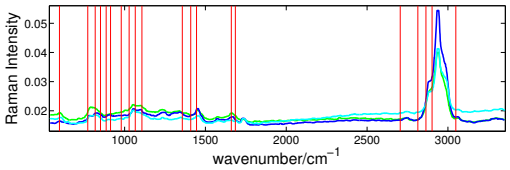

Figure 1. Example Raman spectra from the *PHB* dataset together with 20 discriminative features (vertical lines).

ranking-based feature selection methods. These methods aim to order features in descending order according to their discriminability. In vibrational spectroscopy, genetic algorithms are often used for finding relevant features [3]. Since their application is limited to small-scale problems, embedded methods [6] such as partial least squares [15] and random forests [9] recently received a lot of interest.

In this work, we compare a large list of embedded methods, many of which have not been used before in the context of feature ranking for Raman spectra classification.

2. Methods for Feature Ranking

2.1 Linear Predictors

A major body of feature ranking techniques comprise linear models $f(\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{\theta} + \vartheta_0$, with inputs $\boldsymbol{x} \in \mathbb{R}^d$ and outputs $f(\boldsymbol{x}) \in \mathbb{R}$. The linear dependency between parameters $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_d]^T$ and input dimensions enables to scale features according to their relevance. Large parameters θ_i increase the impact of the i-th feature on the output. An automatic relevance assessment can be accomplished by optimizing the parameter vector $\boldsymbol{\theta}$ with respect to a loss function $\ell(\boldsymbol{X}, \boldsymbol{y})$, where inputs $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]^T$ and outputs $\boldsymbol{y} = [y_1, \ldots, y_n]^T$ both form the training data. Absolute weights $|\theta_i|$ can then be used for feature ranking.

Partial Least Squares In the field of chemometrics, *partial least squares* (PLS) regression is used for a variety of applications [15]. Multivariate inputs $\boldsymbol{X}$ and outputs $\boldsymbol{Y}$ are described by the bilinear factor model

$$\boldsymbol{T} = \boldsymbol{X}\boldsymbol{W}, \quad \boldsymbol{U} = \boldsymbol{Y}\boldsymbol{C} \tag{1}$$

where a mapping $y(\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{B}$ needs to be found that maps through the *low-dimensional* space accessed by $\boldsymbol{W}$ and $\boldsymbol{C}$. Since the parameters stored in $\boldsymbol{B}$ are linearly associated to the features, one can use their absolute values $|b_{ik}|$ as relevance scores for feature i [15]. Summing over all outputs leaves us with one ranking score for each feature (PLS-B). A similar approach uses the projection matrix $\boldsymbol{W}$ [15]. Instead of using a simple sum over absolute values, the components are weighted by the percent of output variance explained by the learned PLS model (PLS-WY).

Regularized Logistic Regression In logistic regression, class probabilities $p(y|\boldsymbol{x}, \boldsymbol{\theta}, \vartheta_0) = \varsigma(y \cdot f(\boldsymbol{x}))$ for outputs $y \in \{-1, +1\}$ are modeled via the sigmoid logistic function $\varsigma(z) = [1 + \exp(-z)]^{-1}$. By optimizing the parameters such that the negative log-likelihood $\ell(\boldsymbol{X}, \boldsymbol{y}) = -\sum_{i=1}^{n} \log p(y_i|\boldsymbol{x}_i, \boldsymbol{\theta}, \vartheta_0)$ is minimized, ranking scores $|\theta_i|$ are obtained. In order to avoid overfitting, a regularized objective can be optimized:

$$\ell_\lambda(\boldsymbol{X}, \boldsymbol{y}) = \ell(\boldsymbol{X}, \boldsymbol{y}) + \lambda\Omega(\boldsymbol{\theta}) \tag{2}$$

where function $\Omega : \mathbb{R}^d \to \mathbb{R}$ penalizes complex models. Ridge logistic regression (L^2-RLR) is achieved for the L^2-norm regularizer $\Omega(\boldsymbol{\theta}) = \frac{1}{2}||\boldsymbol{\theta}||_2^2$. Analogously, the L^1-norm penalty $\Omega(\boldsymbol{\theta}) = |\boldsymbol{\theta}|_1$ leads to sparse logistic regression (L^1-RLR).

L^0-norm Support Vector Machines Linear support vector machines (SVMs) model the relationship $y(\boldsymbol{x}) = \text{sgn}(f(\boldsymbol{x}))$ between inputs $\boldsymbol{x}$ and binary outputs $\boldsymbol{y} \in \{-1, +1\}$, where sgn denotes the signum function. The optimization process can be cast as an instance of regularized optimization problem (2) using L^2-norm penalty $\Omega(\boldsymbol{\theta}) = ||\boldsymbol{\theta}||^2/2$ and the *hinge loss* $\ell(\boldsymbol{X}, \boldsymbol{y}) = \sum_{i=1}^{n} |1 - y_i f(\boldsymbol{x}_i)|_+$, where $|z|_+ = \max(0, 1 - z)$. For the context of feature selection, [14] proposed to use an "L^0"-penalty, which is equal to the number of non-zero parameters in $\boldsymbol{\theta}$. Since this leads to an NP-hard optimization problem, the authors used an iterative scheme for approximating the L^0-SVM. Their approach consists of a sequence of standard SVMs, whose inferred parameters are used to rescale the training data, *i.e.*$\boldsymbol{x}_i \leftarrow \theta_i \cdot \boldsymbol{x}_i$. This procedure of data re-scaling and SVM optimization is repeated until a stopping criterion is met.

dataset name	size	dim	classes
endospores	499	2701	5
vegetative	426	2701	7
PHB	621	2811	5

Table 1. Dataset information

2.2 Ensemble Classifiers

Instead of using a single predictor, *ensemble classifiers* combine several base predictors. They allow for feature selection and ranking if *decision stumps* $h(\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{\theta})$ are employed as base learners [7]. These classifiers split the data according to the information present in a single dimension, which leads to a separation orthogonal to the associated coordinate axis.

Adaptive Boosting In *adaptive boosting* [13], decision stump classifiers are linearly combined in an iterative manner to realize the mapping $y(\boldsymbol{x}) = \text{sgn}\left(\sum_{i=1}^{t} \alpha_i h(\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{\theta}_i)\right)$. In the i-th iteration, parameters α_i and $\boldsymbol{\theta}_i$ are optimized such that an adaptive accuracy criterion is maximized, which focuses on examples that are wrongly classified in the previous iteration. After training is completed, t decision stumps, each working on one dimension, are extracted. This selection process implicitly induces a ranking of dimensions based on their first occurrence in the sequence of t classifiers (BOOST-SEL). Instead of focusing on the first occurrence, all occurrences can be utilized by employing the linear parameters $\boldsymbol{\alpha}$. One sensible ranking criterion for dimension i is to sum up its associated absolute weights $|\alpha_i|$ (BOOST-RANK).

Random Decision Forest As a variant of decision trees, *random forests* [1] aim to partition data in a hierarchical manner. Starting at a root node, a decision stump is learned that distributes the data into two child nodes such that an impurity criterion, *e.g.*, information gain, is optimized. This procedure is repeated in each child node until a node-specific criterion is satisfied. Random forests additionally utilize randomization techniques for robustification. Instead of relying on a single tree, T trees are learned on data that are randomly drawn with replacement. Furthermore, decision stump classifiers are trained with a randomly chosen subset of features. For extracting feature relevance, feature-dependent node statistics can be used. In this work, we used the feature count (RDF-C) and the sum of information gain values (RDF-IG), summed up over all trees.

2.3 Gaussian Processes

In *Gaussian process* (GP) regression, outputs are assumed to be governed by a latent function f and a noise term ε, *i.e.*

$$y(\boldsymbol{x}) = f(\boldsymbol{x}) + \varepsilon \tag{3}$$

Instead of using a parameterized function, f is assumed to be drawn from a GP prior $\mathcal{GP}(m, \kappa)$. The latter can be seen as a generalization to normal distributions over functions and is described by mean function $m(\cdot)$ and covariance function $\kappa(\cdot, \cdot)$. Assuming Gaussian noise, inference for the most likely output y_*, given a previously unseen input $\boldsymbol{x}_*$, is analytically tractable [11]. In GP classification, assumption (3) is not justified due to the discrete nature of output labels. In practice, we can either perform *label regression*, or account for the discrete nature by changing the noise model. For the latter, exact inference is intractable and approximation methods such as *expectation propagation* are required [11].

Feature ranking can be achieved by employing a covariance function that parameterizes each dimension separately, *e.g.*, $\kappa(\boldsymbol{x}, \boldsymbol{x}') = \theta_0^2 \exp\left(-\boldsymbol{x}^T \Lambda \boldsymbol{x}'/2\right)$, where $\Lambda = \text{diag}(\theta_1^2, \ldots, \theta_d^2)$. Hyperparameter optimization can be done by maximizing the data likelihood with respect to $\boldsymbol{\theta}$. Since this is prone to overfitting, we follow [10] and choose the parameters on the optimization path that lead to the maximum *leave-one-out* (loo) probability. While this can be analytically calculated [11] in the case of label regression (GPR-ARD), we use the cavity distribution similar to [10] for approximate loo estimation in GP classification (GPC-ARD). Finally, parameters $|\theta_i|$ are used as a ranking score for the i-th feature.

3. Experiments

Implementation Details For comparison, we used three different bacteria datasets listed in Table 1, which were measured by a micro-Raman setup. For feature ranking, the data is normalized such that each dimension has zero mean and unit standard deviation.

For PLS regression, Matlab's `Statistics Toolbox` is employed. We used `LIBLINEAR` [4] for L^0-SVM and [8] to solve L^1-norm logistic regression. For all regularized objectives, we set $\lambda = 1.0$. Boosting was trained using the `MultiBoost` package [2] using $t = d$ weak learners, where d denotes data dimensionality. For random forests, $T = 100$ trees were employed, each using a third of all training data and 100 features per node. For GP classification, the code distributed along [11] was utilized. To speed up GP learning, we trained 30 independent predictors using a random training subset (50 per class for *PHB* and *endospores*, 30 per class for *vegetative*) instead of a single GP predictor. Binary classifiers were extended to multi-classification using the one-vs-rest scheme.

		number of retained relevant features					
dataset	method	10	20	50	100	200	500
endospores	RF-C	92.1	*94.9*	*98.7*	*99.2*	*99.3*	**100.0**
	RDF-IG	92.1	*95.3*	*98.8*	*99.2*	*99.1*	**100.0**
	BOOST-SEL	**98.5**	99.6	**99.8**	**100.0**	**100.0**	**100.0**
	BOOST-RANK	**98.5**	99.6	**99.8**	**100.0**	**100.0**	**100.0**
	PLS-B	68.8	90.0	*97.1*	*99.8*	*99.7*	**100.0**
	PLS-WY	83.2	91.6	*96.9*	*99.3*	*99.8*	**100.0**
	L^0-SVM	*98.2*	**99.8**	99.4	*99.2*	*98.4*	**100.0**
	L^2-RLR	31.1	77.4	*98.3*	*99.6*	**100.0**	**100.0**
	L^1-RLR	*97.8*	*98.2*	*98.7*	*99.0*	*99.4*	**100.0**
	GPC-ARD	*95.9*	99.3	99.3	*99.8*	**100.0**	**100.0**
	GPR-ARD	94.1	*97.5*	99.5	**100.0**	**100.0**	**100.0**
vegetative	RDF-C	85.3	*91.6*	94.7	96.3	*98.4*	99.6
	RDF-IG	86.6	*92.1*	94.6	96.3	98.9	99.6
	BOOST-SEL	*96.1*	98.6	*98.8*	98.6	**99.6**	**99.8**
	BOOST-RANK	*96.1*	98.2	**98.9**	98.4	**99.6**	**99.8**
	PLS-B	40.9	82.6	96.3	98.1	98.9	99.6
	PLS-WY	73.2	*90.3*	*96.8*	98.0	99.4	99.6
	L^0-SVM	**96.5**	**99.2**	*98.8*	98.5	*97.8*	99.3
	L^2-RLR	43.4	86.4	*97.7*	98.7	99.0	99.6
	L^1-RLR	*94.2*	98.8	*98.5*	98.6	*98.2*	99.3
	GPC-ARD	92.0	96.4	*97.9*	98.7	*98.5*	99.6
	GPR-ARD	87.9	96.8	*98.7*	**99.1**	99.3	99.6
PHB	RDF-C	87.7	90.4	*93.5*	*95.0*	94.0	97.1
	RDF-IG	80.7	83.3	94.2	*94.6*	94.2	96.8
	BOOST-SEL	*93.8*	**97.0**	96.1	*94.7*	**96.0**	97.5
	BOOST-RANK	**94.3**	*96.7*	**97.9**	**95.8**	**96.0**	97.5
	PLS-B	62.2	78.6	87.2	*93.1*	94.2	97.6
	PLS-WY	57.1	64.7	81.9	*92.4*	94.4	97.7
	L^0-SVM	*93.7*	*96.2*	*92.9*	*93.1*	94.0	**98.0**
	L^2-RLR	70.9	81.6	*93.7*	*94.8*	94.9	*97.1*
	L^1-RLR	*83.6*	*84.4*	*89.4*	*90.4*	94.2	*97.1*
	GPC-ARD	*88.9*	*93.1*	95.9	*94.4*	95.8	97.4
	GPR-ARD	85.1	*95.2*	96.2	*93.6*	95.6	97.1

Table 2. Performances on all datasets.

For analyzing the feature ranking ability of all embedded methods, we employed 10-fold cross-validation. Performance was assessed using a GP label regression scheme.

Results Table 2 displays the classification performances (measured in average recognition rate) when projecting the data onto the first few discriminative features. While all methods converge to very similar classification rates when using more dimensions, there is a clear performance difference among the first 10, 20, and 50 features. It is apparent that both boosting based ranking methods achieved the overall best performance, followed by L^0-SVM. Sparse logistic regression, Gaussian process based methods, as well as node statistics from random forest have a moderate ability in extracting relevant features. Ridge logistic regression and the PLS based ranking approaches seem to fail in finding discriminative features.

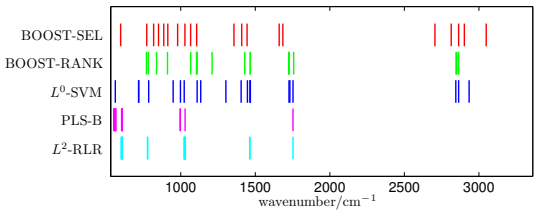

Figure 2. The 20 most discriminative features for the first fold of the *PHB* dataset.

The clear superiority of boosting can be explained by its iterative optimization scheme. At each iteration, only the feature leading to the highest increase in accuracy is selected. Neighboring features, which are likely to highly correlate, are usually not considered in this step. Sparse linear models such as zero-norm SVMs and sparse logistic regression select variables in a similar vein [16]. Weight vectors associated to *all but one* highly correlated features are usually set equal to zero. For L^2-norm regularized methods such as ridge regression and Gaussian process methods, highly correlated variables often receive similar ranking scores. This effect is even more pronounced in PLS, where no regularization term is employed.

Figure 2 supports this reasoning. For the three best and two worst performing methods, the 20 most relevant features inferred from the first fold of the *PHB* dataset are shown. While boosting and zero-norm SVM select wavenumbers from the whole range, features from PLS and ridge logistic regression cluster around a few frequencies. However, note that no method votes for features from the *Raman silent* region 1800 – 2700 cm^{-1}, which is physically plausible since nearly no biological molecule vibrates in this frequency range [5].

4. Conclusions and Future Work

This work focused on finding relevant features for high-dimensional Raman spectroscopy data. A multitude of methods, including linear and non-linear classifiers, are compared on three bacteria datasets. Our analysis shows that boosting and zero-norm SVMs are suitable for extracting highly discriminative features, substantially outperforming other established methods such as ranking methods based on partial least squares and random forest. We plan to extend this study using larger, more complex Raman spectra databases. Furthermore, an adaption of all methods for finding relevant coherent intervals would be interesting.

Acknowledgments

We would like to thank Stephan Stöckel and Valerian Ciobotă for capturing the Raman spectra datasets. This work was partially funded by the TMBWK ProExzellenz project "MikroPlex" (PE113-1).

References

[1] L. Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, October 2001.

[2] N. Casagrande. Multiboost: An open source multi-class adaboost learner, 2005. http://iro.umontreal.ca/ casagran/multiboost/.

[3] S. Duraipandian, W. Zheng, J. Ng, J. J. Low, A. Ilancheran, and Z. Huang. In vivo diagnosis of cervical precancer using raman spectroscopy and genetic algorithm techniques. *Analyst*, 136:4328–4336, 2011.

[4] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. *J. Mach. Learn. Res.*, 9:1871–1874, 2008.

[5] K. Fujita, H. Yamakoshi, K. Dodo, M. Sodeoka, A. Palonpon, J. Ando, M. Okada, and S. Kawata. Raman imaging of alkyne as a small tag for biological molecules. In *SPIE Photonics West*, 2012.

[6] I. Guyon and A. Eliseeff. An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182, 2003.

[7] W. Iba and P. Langley. Induction of one-level decision trees. In *Proc. ICML*, 1992.

[8] S.-J. K. K. Koh and S. Boyd. An interior-point method for large-scale l1-regularized logistic regression. *J. Mach. Learn. Res.*, 8:1519–1555, 2007.

[9] B. Menze, B. Kelm, R. Masuch, U. Himmelreich, P. Bachert, W. Petrich, and F. Hamprecht. A comparison of random forest and its gini importance with standard chemometric methods for the feature selection and classification of spectral data. *BMC Bioinformatics*, 10:213–228, 2009.

[10] Y. A. Qi, T. P. Minka, R. W. Picard, and Z. Ghahramani. Predictive automatic relevance determination by expectation propagation. In *Proc. ICML*, pages 85–, 2004.

[11] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2005.

[12] P. Rösch, M. Harz, K.-D. Peschke, O. Ronneberger, H. Burkhardt, and J. Popp. Identification of single eukaryotic cells with micro-raman spectroscopy. *Biopolymers*, 82(4):312 – 316, 2006.

[13] R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Mach. Learn.*, 37(3):297–336, December 1999.

[14] J. Weston, A. Elisseeff, B. Schölkopf, and M. Tipping. Use of the zero-norm with linear models and kernel methods. *J. Mach. Learn. Res.*, 3:1439–1461, 2003.

[15] S. Wold, M. Sjöström, and L. Eriksson. PLS-regression: a basic tool of chemometrics. *Chemom. Intell. Lab. Syst.*, 58:109–130, 2001.

[16] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *J. R. Statist. Soc. B*, 67(2):301–320, 2005.