ACTA UNIVERSITATIS AGRICULTURAE ET SILVICULTURAE MENDELIANAE BRUNENSIS

Volume LX 8 Number 2, 2012

# TIME SERIES CLASSIFICATION USING K-NEAREST NEIGHBOURS, MULTILAYER PERCEPTRON AND LEARNING VECTOR QUANTIZATION ALGORITHMS

J. Fejfar, J Šťastný, M. Cepl

**Received: November 30, 2011**

FEJFAR, J., ŠŤASTNÝ, J., CEPL, M.: *Time series classification using k-Nearest neighbours, Multilayer Perceptron and Learning Vector Quantization algorithms.* Acta univ. agric. et silvic. Mendel. Brun., 2012, LX, No. 2, pp. 69–72

We are presenting results comparison of three artificial intelligence algorithms in a classification of time series derived from musical excerpts in this paper. Algorithms were chosen to represent different principles of classification – statistic approach, neural networks and competitive learning. The first algorithm is a classical k-Nearest neighbours algorithm, the second algorithm is Multilayer Perceptron (MPL), an example of artificial neural network and the third one is a Learning Vector Quantization (LVQ) algorithm representing supervised counterpart to unsupervised Self Organizing Map (SOM).
After our own former experiments with unlabelled data we moved forward to the data labels utilization, which generally led to a better accuracy of classification results. As we need huge data set of labelled time series (a priori knowledge of correct class which each time series instance belongs to), we used, with a good experience in former studies, musical excerpts as a source of real-world time series. We are using standard deviation of the sound signal as a descriptor of a musical excerpts volume level.
We are describing principle of each algorithm as well as its implementation briefly, giving links for further research. Classification results of each algorithm are presented in a confusion matrix showing numbers of misclassifications and allowing to evaluate overall accuracy of the algorithm. Results are compared and particular misclassifications are discussed for each algorithm. Finally the best solution is chosen and further research goals are given.

classification, k-Nearest Neighbours, Multilayer Perceptron, Learning Vector Quantization

We are comparing three well-known supervised learning algorithms results in this paper. This experiment follow-up a paper (Fejfar *et al.*, 2011), where unsupervised learning algorithms results (in time series clustering problem) comparison was given. Algorithms in this paper, compared to foregoing one, utilise the supervised learning paradigm, that comprises prior knowledge of data classes (data labels), which generally increases the accuracy of classification algorithms. Data and methods, briefly described in the next chapter, are the same so it enables comparison of experiments results. This experiment also serve as a validation of our LVQ algorithm implementation, which was supported with the grant mentioned in the beginning of the paper. This methods can be used in augmented reality algorithms (Šťastný, 2011) or in modelling and simulation of PID controller (Koprda, 2011).

## METHODS AND RESOURCES

Methods used for the algorithms comparison in this paper are similar to the methods used in the paper (Fejfar *et. al.,* 2011) as far as using Confusion matrix in this experiment rather than Matching matrix in previous one. Data are labelled sound excerpts from the Magnatagatune database (Law, Von Ahn, 2009). We are using “heavy” and “silence” labelled instances.

# RESULTS

## K-nearest neighbours

K- nearest neighbours algorithm belongs to the Instance based learning algorithms, as it estimates the category of unknown instance, in accordance with labels of its (k) nearest neighbours. It is very simple and straightforward algorithm demonstrating label knowledge utilisation. We make use of WEKA implementation configured with following command:

```
weka.classifiers.lazy.IBk -K 3 -W 0 -A
...”weka.core.neighboursearch.LinearNNSearch -A
...\”weka.core.EuclideanDistance -R first-last\””
```

Results are represented in Tab. I, showing the occurrence of 4 mistakes out of 285 instances, giving the accuracy of the algorithm to 98.6 %. This very good result is evincible as data vectors are well separable (Fejfar *et al.*, 2011).

I: *k-nearest neighbours (k=3) confusion matrix*

| | | Data labels | |
|---|---|---|---|
| | | **“heavy”** | **“silence”** |
| **Algorithm result** | **“heavy”** | 215 | 2 |
| | **“silence”** | 2 | 66 |

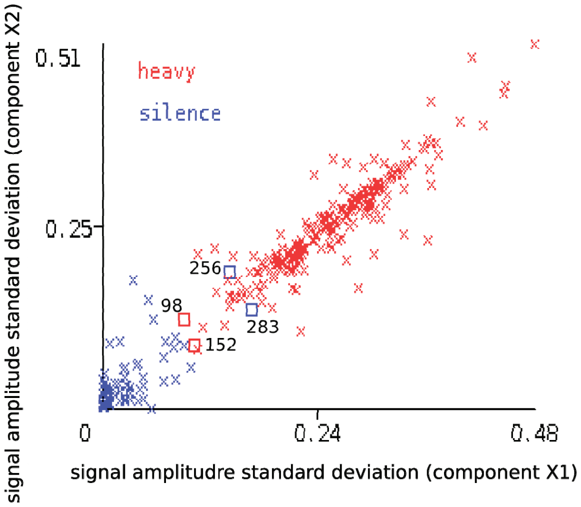
We can obtain even more interesting results with data visualization. For this purpose we can reduce the number of characteristic vector dimension to 2. The resulting diagram is shown in Fig. 1. Instances classified correctly as “heavy” are marked with red crosses (there are 215 of such instances), instances correctly classified as silence are marked with blue crosses (66 instances), instances incorrectly classified as heavy (they are labelled silence) are marked with blue circle (2 instances) and instances incorrectly classified as silence (they are labelled heavy) are marked with red circle (2 instances).

Factually we can see misclassified instances in Tab II.

There are two types of misclassification: true negatives and false positives. Furthermore we can find the path to the particular file in Magnatagatune database. The path starts with “magnatagatune/mp3/” and continue with subdirectory (0-f) ending with the name of the file.

## Multi-layer Perceptron

Multi-layer Perceptron is an example of artificial neural network. The principle is much more complicated, than in the previous case, and is described with economical application in (Štencl, Šťastný, 2009). We used GNU Octave implementation due to its compatibility with MATLAB as well as its GNU GPL licence, enabling source code review. It is necessary to add MLP



1: *Visualisation of the kNN results*

II: *kNN misclassified instances*

| Instance number | Label | Classification | Name of the recording |
|---|---|---|---|
| 98 | heavy | silence | 3/mandrake_roothe_seventh_mirror-04-put_your_money_where_your_mouth_is-59-88.mp3 |
| 152 | heavy | silence | b/rebel_rebel-explode_into_space-07-articial_kid-233-262.mp3 |
| 256 | silence | heavy | 9/wicked_boy-chemistry-06-ten years-0-29.mp3 |
| 283 | silence | heavy | 4/myles_cochran-marginal_street-15-such_a_sunny_day-407-436.mp3 |

functionality to GNU Octave with “nnet” package, which is partially compatible with MATLAB Neural Network Toolbox.

The input file (.csv) includes feature vector components and a label in the numerical code (1 for heavy, 0 for silence):

```
0.222537717091171,0.1905727531251785,0.189240
5989421166,1
0.215563379493032,0.2178567180582369,0.217267
3918631148,1
0.254683487934332,0.2814293423145541,0.315932
0592857192,1;
```

We can load the file in GNU Octave by executing commad:

```
mData = load(“export/heavy_silence_3d_std_octave.
csv”);
```

We need to split data into Training, Validating and Testing sets. It can be done with command “subset”:

```
[mTrain, mTest, mVali] = subset(mData’, 1,...
output_neuron_count, 1/3, 1/6);
```

This task can be done by MPL generally in two configurations. In first case we can have one output neuron with output value 1 if classifying “heavy” or 0 if classifying “silence” (also it is possible to gain value 0,5 if we use linear transitional function). The second configuration consists of two output neurons. One is representing class “heavy” and the second is representing class “silence”. Output 1 means that neuron is active and output 0 stands for passive neuron.

In the next step we split the data into the input and the target sets. Input data are submitted to the network and output value of the network is requested to be same as output data. We can split data and retrieve minimal and maximal value of input data with commands:

```
Train.P=mTrain(1:end-output_neuron_count,
:) %tr. input
Train.T=mTrain(end, :); %train target
Test.P=mTest(1:end-output_neuron_count,
:); %test input
Test.T=mTest(end, :); %test target
VV.P = mVali(1:end-output_neuron_count,
:); %val. input
VV.T = mVali(end, :); %validation target
mMinMaxElements = min_max(Train.P);
```

In the next step we create MPL network, train the network to fit output data values and test its performance on still unused data. At the end we save results of the network in the (.csv) file:

```
topology = [2 1];
MLPNet = newff(mMinMaxElements, topology,
{“tansig”, “purelin”},...
“trainlm”, “not used”, “mse”);
net = train(MLPNet, Train.P, Train.T, [], [], VV);
%saveMLPStruct(net, “MLPNet.txt”);
simOut = sim(net, Test.P);
csvwrite(‘MLPvysledek.csv’, [mTest’ simOut’]);
```

We can see resulting confusion matrix in Tab. III.

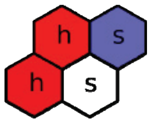
III: *MPL confusion matrix*

| | | **Data labels** | |
|---|---|---|---|
| | | **“heavy”** | **“silence”** |
| **Algorithm result** | **“heavy”** | 72 | 0 |
| | **“silence”** | 1 | 22 |

There are only 95 recordings which corresponds to 1/3 of the amount of the testing data. There is 1 misclassified recording out of 95, which gives the accuracy of 98.95 %. Because the data for training, validating and testing are randomly selected, we can suppose, that results on another selected data will be the same.

## Learning Vector Quantization

Learning Vector Quantization algorithm is described in (Kohonen, 2001). We used our own implementation in C++ because this algorithm is not found in any Open Source software library or framework. Several topologies were tested and topology with 4 neurons was finally selected. We can see selected topology in Fig 2. It consists with four nodes (neurons) organized into hexagonal grid. Two of them (with letter “h”) are representing recordings classified as “heavy” and another two nodes (with letter “s”) are representing recordings classified as “silence”. After the training process the results of algorithm are compared with real situation and colours are added into the diagram. Red colour is for recordings labelled as “heavy” and blue colour is for recordings classified as “silence”.



2: *LVQ Topology*

We can see classification results in Tab. III. showing the occurrence of 7 mistakes out of 285 instances giving the accuracy of the algorithm to 97.5 %.

IV: *LVQ confusion matrix*

| | | **Data labels** | |
|---|---|---|---|
| | | **“heavy”** | **“silence”** |
| **Algorithm result** | **“heavy”** | 212 | 2 |
| | **“silence”** | 5 | 66 |

There are the same misclassified instances as in the kNN case (4 instances) and 3 more instances as show Tab. V. Recordings 13 and 44 have the common author as misclassified recordings 98 and 152 in Tab. II.

V: *kNN misclassified instances*

| Instance number | Label | Classification | Name of the recording |
|---|---|---|---|
| 13 | heavy | silence | 3/mandrake_root-the_seventh_mirror-01-kings_ of_the_desert-0-29.mp3 |
| 44 | heavy | silence | b/rebel_rebel-explode_into_space-02-we_ are_the_ future-146-175.mp3 |
| 109 | heavy | silence | 7/rocket_city_riot-pop_killer-05-im_gonna_ make_you_bleed-0-29.mp3 |

## DISCUSSION

We presented the comparison of three classification algorithms. Each of them utilises different learning principle. In the case of kNN it is Instance based learning, MLP is example of the artificial neural network and LVQ algorithm is using Competitive learning strategy. Despite of different principles all the algorithms give very similar results, which denote consistency of demonstrated experiment.

## SUMMARY

The experiment show usability of presented algorithm in a classification of data such as time series derived from musical excerpts. Another benefit of presented experiment is our new LVQ implementation validation. Although it acquired slightly worse accuracy than kNN or MLP it can be used with our SOM implementation together to form semi-supervised learning algorithm for Kohonen network topology. This semi-supervised learning algorithm is presented in (Fejfar, 2011).

Acknowledgement

This paper is supported by IGA project 39/2011.

## REFERENCES

FEJFAR, J., 2011: *Application of Modern Methods for Sound Data Classification*. Ph.D. thesis. Brno.

FEJFAR, J., MOTYČKA, A., FILÍPEK, Š., 2011: Algorithms for time series clustering comparison. NAUN/IEEE.AM International Conferences, WSEAS.

LAW, E., VON AHN, L., 2009: Input-agreement: A New Mechanism for Data Collection using Human Computation Games. Proc. Of CHI, Boston, Massachusetts, USA. ACM press 978-1-60558-247-4, pp. 1197–1206.

KOHONEN, T., 2001: Self-Organizing Maps. Secaucus, NJ, USA: Springer-Verlag New York, Inc. ISBN 3540679219.

KOPRDA, Š., BALOGH, Z., TURČÁNI, M., 2011: Modeling and comparison of fuzzy PID controller with PSD regulation in the discrete systems. INTERNATIONAL JOURNAL OF CIRCUITS, SYSTEMS AND SIGNAL PROCESSING. ISSN 1998-4464, Vol. 5, Issue 5, pp. 496–504.

ŠTENCL, M., ŠŤASTNÝ, J., 2009: Advanced approach to numerical forecasting using artificial neural networks. Acta Universitatis agriculturae et silvicultrae Mendelianae Brunensis, sv. 6, č. 2, pp. 297–304, ISSN 1211-8516.

ŠŤASTNÝ, J., PROCHÁZKA, D., KOUBEK, T., LANDA, J., 2011: Augmented reality usage for prototyping speed up. Acta Universitatis agriculturae et silviculturae Mendelianae Brunensis sv. LIX, č. 2, s. 353–360. ISSN 1211-8516.

Address

Ing. Jiří Fejfar, doc. RNDr. Ing. Jiří Šťastný, CSc., Ing. Miroslav Cepl, Ústav informatiky, Mendelova univerzita v Brně, Zemědělská 1, 613 00 Brno, Česká republika, e-mail: jiri.fejfar@mendelu.cz