Proceedings

Comparison of machine learning algorithms for processing of original data of electronic nose for analysis of biological samples of humans and animals †

Tatiana Kuchmenko [1,*], Anastasiia Shuba [1], Ruslan Umarkhanov[1], Vladimir Krylov[2]

[1] Voronezh State University of Engineering Technologies; tak1907@mail.ru
[2] High School of Economics, Nizhny Novgorod; vkrylov@hse.ru
* Correspondence: tak1907@mail.ru; Tel.: +79204227725

† Presented at the title, place, and date.

Abstract: The study aims to build a mathematical model capable of classifying biosamples with minor errors into groups corresponding to clinical diagnoses by the original output data of the mass-sensitive sensor array. One hundred forty-four calves were clinically and laboratory examined and divided by the health of respiratory organs into three groups. A sample of nasal secretion was taken from each calf. The gaseous phase over samples was measured using an array of 8 mass-sensitive sensors with solid-state nanostructured coatings in the open detection cell. During the sorption and desorption of volatile substances excreted from the samples, the sensor responses were recorded in software and then processed by un-, semi – and supervised machine learning methods. In total, 50 algorithms for processing sensor data were studied, including t-SNA, self-learning model DBSCAN, Yarovsky algorithm, BOSSVS, SAXVSM, LearningShapelets, MultivariateClassifier. The nonlinear transformation of the original sensor data was used in order to obtain the simplest two-dimensional manifold on which all data points will be located separately, such as Locally Linear Embedding, Local Tangent Space Alignment, Hessian Eigenmapping, Modified Locally Linear Embedding, Isomap, Multi-dimensional Scaling, Spectral Embedding, t-distributed Stochastic Neighbor Embedding. The supervised machine learning models using the Dynamic Time Warping metric of similarity between two-time series and the k-NN algorithm for classification achieved a correct classification accuracy equal to 0.83.

Keywords: sensor array; machine learning; original data transformation; classification; diagnostics; nasal secretion.

Citation: Kuchmenko, T.; Shuba, A.; Umarkhanov, R.; Krylov V. Comparison of machine learning algorithms for processing of original data of electronic nose for analysis of biological samples of humans and animals. Chem. Proc. 2021, 3, x. https://doi.org/10.3390/xxxxx

Published: date

1. Introduction

Here we will look at developing a pipeline for stepwise processing of data coming from the eight sensors in the portable electronic nose. Our goal is to obtain a chain of software modules, the sequential application of which to raw data from sensors at the input generates the output number as the most plausible class for this data. Therefore, we have the classic classification problem. For the solution, we will construct a trainable model using the supervised learning method. The result is the processing pipeline in inference mode shown in Figure 1a. The main requirements for this pipeline are computational compactness and speed of classification since it focused on integration into a separate diagnostic device, for example, a portable e-nose with an open detection cell.

The choice of a classifier model and its training is a separate task, the solution of which is a processing pipeline that includes data visualization tools, modules for preliminary processing of transformation, filtering, and normalization of data, a trained classifier model and tools for testing and evaluating the quality of classification.

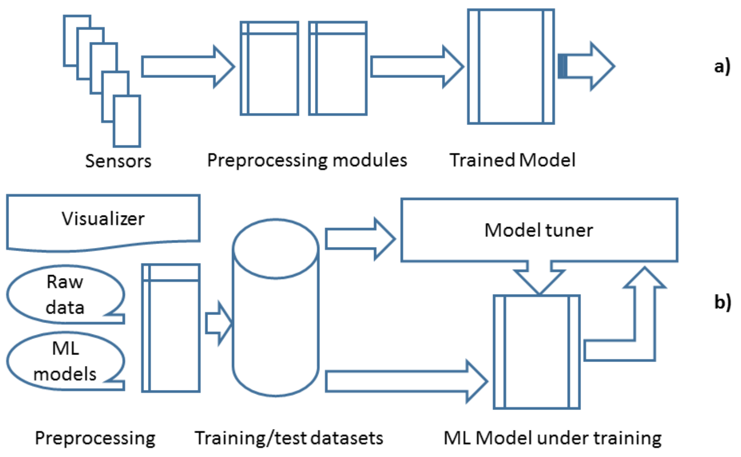

Figure 1. The processing pipeline in inference mode (a) and training mode (b).

Figure 1b shows this pipeline in training mode. The main requirement for this mode is the convenience of replacing models and assessing the model quality.

2. Methods and objects

The data from portable e-nose with eight piezoelectric sensors covered by nanostructured coatings [1] was used. The objects of analysis were samples of nasal secretions of calves (n=144) to diagnose the respiratory disease (rhinitis, bronchitis, pneumonia). Each calf was examined by clinical and laboratory methods, including Wisconsin respiratory scoring chart, induced cough. The calves were divided into three classes (“sick” – calves with pneumonia and trachea bronchitis (n=35), “subclinical” – calves with first sign of respiratory disease, rhinitis, bronchitis (n=69), “normal”- calves without sign of respiratory disease (n=30)). For each calf, the sample of nasal secretion was collected and analyzed by portable e-nose. The measurement mode was described in work [1]. Some of the samples from calves with pneumonia were investigated by bacteriological, molecular genetic methods for common viruses and bacteria of respiratory disease in calves. We suggest that the volatile profile of nasal secretion with respiratory disease will be different depending on the degree and depth of damage to the respiratory system.

3. Developing a processing pipeline in training mode

The first step here is an exploratory analysis of the existing dataset of raw data from the e-nose sensors and developing a classification concept. According to the measurement mode of gas phase analysis using e-nose in each experiment, the sensors generate eight sequences of numbers (up to 200 per sensor). Each experiment can be assigned to one of three classes with labels: 1, 0, -1 (“sick”, “subclinical”, “normal”). Figure 2 shows an example of plots for data from sensors called "Chrono-Frequency-Gram" (CFG).

According to the measurement mode, CFG contains three stages of sensor operation:

Time interval from 0 to 80 seconds – sorption; the volatile compounds, excreted from secretion sample, enter the pre-sensory volume into detection cell and interact with coatings;

Time interval from 80 to 85 seconds – depressurization, uninformative process;

Time interval from 85 to 200 seconds – desorption, the volatile compounds is spontaneously removed from the sensor coatings and detection cell.

The following describes the processing of data obtained only at the first stage. Analysis of the raw data shows that the use of the first CFG values can introduce significant errors due to their strong noisiness for many experiments, and therefore, for further processing, values indexed from 13 to 73 are used only. The noise of the first 13 seconds connects with the non-automatic moving of the device with an open detection cell.

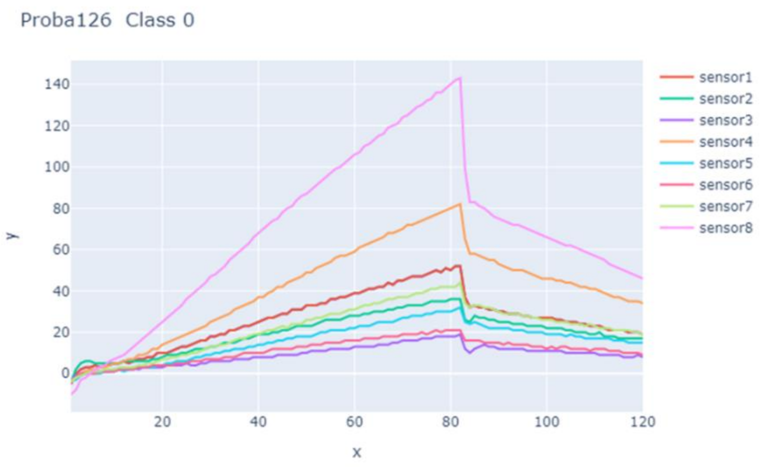

Figure 2. Chrono-Frequency-Gram's from the eight e-nose sensors for experiment #126. The classification label is 0. The horizontal time axis x is the seconds, and the vertical axis y is the QCM frequency deviation in hertz.

We will associate with these data from each sensor a numerical sequence of dimension 60. Considering physical specifics of these data, coming from microbalances, as a monotonic display of mass accumulation on the sorbent over time, we will use not the sequence of frequency deviations itself but the calculated derivative of this sequence. We will calculate the derivative of a noise-distorted sequence using the method of A. Savitzky, M. J. E. Golay published in [2]. Figure 3 shows the CFG derivative sequences for all sensors of the same experiment as in Figure 2.

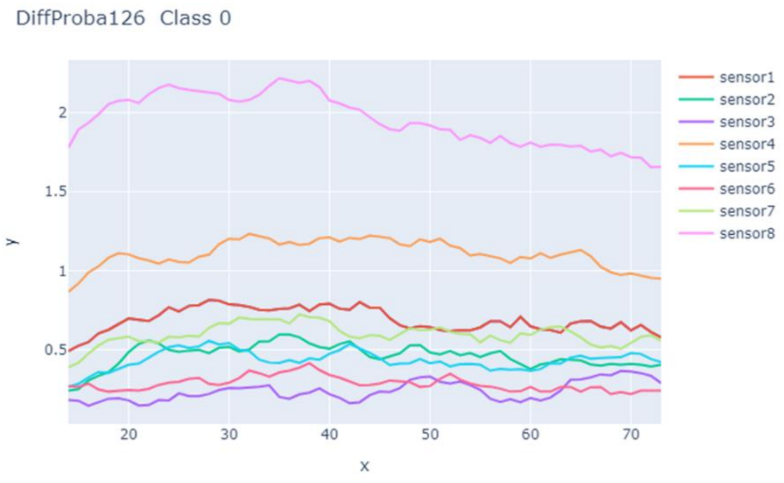

Figure 3. Sensors sequence derivative plots for sample #126.

The sequences of values of the derivation of the signal from the sensors will be further used as input data. Their dimension is also 60 and they should be considered precisely as time series, but not just vectors of the same dimension. Their values do not mean individual properties: they cannot be rearranged, cannot be normalized and scaled separately, and therefore most of the feature engineering methods cannot be used for data preprocessing.

Following the main stages of exploratory data analysis, we visualize the existing dataset. To reduce the dimension, we will concatenate the time series from all sensors for each experiment and find the projection of the resulting 480-dimensional vectors onto the two-dimensional space by finding a suitable two-dimensional manifold UMAP - Uniform Manifold Approximation and Projection for Dimension Reduction [3].

Figure 4 shows labeled “sick” (red) and “normal” (green) data points. Various algorithms can be used to classify time series. Since in the literature we did not find examples of algorithms that work well for the classification of short series like ours, it was decided to conduct a study of many machine learning (ML) algorithms based on different approaches. For this, two fairly powerful libraries for machine learning on time series were chosen: SKTIME, developed by the Alan Turing Institute [4] and pyts - A Python Package for Time Series Classification [5]. These libraries can work with the convenient and popular representation of Pandas Dataframe data but require different formats for the training dataset.

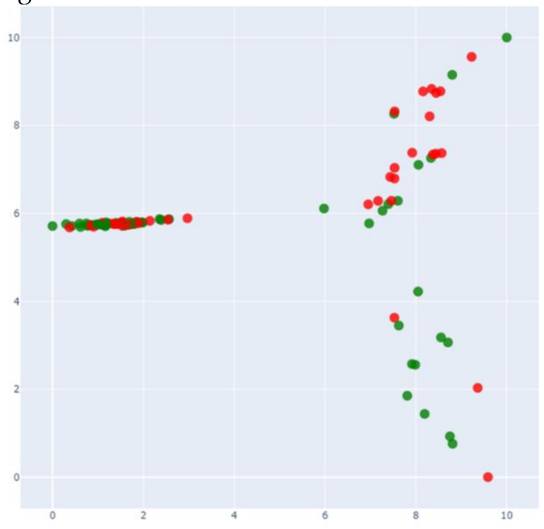

Figure 4. 2-D data representation using UMAP technology.

SKTIME uses a specific representation of data in the form of a Nested Pandas Dataframe. This view requires the formation of a table in the form of a Pandas Dataframe in which experiments are placed in rows (we have 144), in columns - time series corresponding to one sample (we have 8 sensors - 8 Pandas Dataframe columns), and in each cell of 144x8 Pandas Dataframe are placed the time series themselves obtained from the sensor signals. We will here use the derivative values as a time series with 60 values for this series.

Pyts uses Numpy ndarray 3-D array representation of data. The first index shows the number (starting from zero) of the sample (we have from 0 to 143), the second index shows the sensor number (from zero to 7), the third index shows the time stamp in the time series (we have from 0 to 59). For training, class labels are placed in a separate array of 144, whose values are 1, 0, -1. Thus, the preprocessing modules in the training mode form the data sets necessary for training the model.

4. Machine learning model development.

Next, we will list all the ML classifier models that have been tested in this pipeline and give the results of their work. Then, after discussing the results, a model will be described that, rather unexpectedly, gave good results (Table 1). Considering the obtained results of training various ML models developed specifically for the classification of time series, it can be seen that the classification accuracy turns out to be very low. Of course, the problem was solved on too little dataset. The number of experiments (144) is not enough to train a complex model. However, we believe that the main reason for this quality of models is the small length of the series and their insufficient diversity. Both of these shortcomings are caused by the physics of sensors and cannot be improved at the level of data preprocessing. The possibility of improving the quality of the classification can only be found in finding the best models of the trained classifiers. This conclusion is confirmed by a significant improvement in accuracy when applying the model shown in the last row of the Table 1.

Table 1. The machine learning classifier models tested in the pipeline and accuracy of its classification.

Method	Module	Accuracy
Bag-Of-Symbolic Fourier Approximation -Symbols in Vector Space	pyts.classifier.BOSSVS	0.5
k-nearest neighbors classifier	pyts.classifier.KNeighborsClassifier	0.5
Symbolic Aggregate approximation and Vector Space Model.	pyts.classifier.SAXVSM	0.4
Learning Time-series Shapelets	pyts.classifier.LearningShapelets	0.4
Classifier wrapper for multivariate time series	pyts.classifier.MultivariativeClassifier	0.6
Applies estimators to columns of an array or pandas DataFrame	sktime. ColumnEnsembleClassifier	0.5
Bag-Of-Symbols Ensemble technic	BOSSEnsemble	0.6
Tree ensemble method, referred to as time series forest	sktime.TimeSeriesForestClassifier	0.6
k-NN DTW Similarity	1-NNDTWClassifier	0.7

This machine learning model is a simple k-NN classifier; for classification, an object is assigned to the class that is most common among the k neighbors of a given element, whose classes are already known. We used the model of a single neighbor k = 1, which made it possible to work with such a small amount of training dataset. However, instead of the common metrics for determining the distance between time series, we used a special metric for such a series called dynamic time warping (DTW). It should be noted that such a model was studied earlier [6] but proved to be insufficiently effective. Our task turned out to be the only one sufficiently accurate to implement the processing of signals from e-nose sensors in model inference mode.

A feature of calculating the DTW metric is the displacement of adjacent samples of the sequences during the calculation so that they use the most significant similarity of curves with specific restrictions and rules. To speed up the calculation, instead of the exact value of the metric, the definition of its lower bound is often used. We did this also in our work. Below is a fragment of the main module of the classifier model based on the use of the lower bound LB Keogh [7].

```python
def knn(train,test,w):
    preds=[]
    for ind,i in enumerate(test):
        min_dist=float('inf')
        closest_seq=[]
        #print ind
        for j in train:
            if LB_Keogh(i[:-1],j[:-1],5)<min_dist:
                dist=DTWDistance(i[:-1],j[:-1],w)
                if dist<min_dist:
                    min_dist=dist
                    closest_seq=j
        preds.append(closest_seq[-1])
    return classification_report(test[:,-1],preds)
```

Below is a screenshot of evaluating the quality of this model on "sick"/"normal" classes if use catted training dataset contains 75 experiments only:

```
              precision    recall  f1-score   support

         0.0       0.67      1.00      0.80         6
         1.0       1.00      0.25      0.40         4

    accuracy                           0.70        10
   macro avg       0.83      0.62      0.60        10
weighted avg       0.80      0.70      0.64        10
```

The precision of kNN classification model is 83 %, which is appropriate for screening diagnostic tasks, mainly using raw sensor data.

4. Conclusion

In this work, we have built a data processing pipeline from an eight-sensor e-nose, which allows us to input raw data from sensors, transform them into time series of one minute in duration and classify them into two classes: "normal" or "sick", as well as to separate the subclinical case when reliable decision-making on data is an unacceptable risk. Using a machine learning model with f a special time series comparison metric made it possible to train the model to an accuracy of 83% even on 75 experiments.

Author Contributions: Conceptualization, T.A.; methodology, V.K.; software, V.K.; validation, R.U., A.S.; investigation, A.S., R.U., T.K.; data curation, V.K.; writing—original draft preparation, V.K.; writing—review and editing, A.S.; visualization, V.K.; supervision, T.A.; project administration, R.U.; funding acquisition, T.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Russian Science Foundation, grant number 18-76-10015..

Institutional Review Board Statement: The study was conducted according to the guidelines of the Declaration of Helsinki, and approved by Ethics Committee of Voronezh State University of Engineering Technologies (protocol code № 2 from 21 February 2021).

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Kuchmenko, T.; Shuba, A.; Umarkhanov, R.; Chernitskiy, A. Portable Electronic Nose for Analyzing the Smell of Nasal Secretions in Calves: Toward Noninvasive Diagnosis of Infectious Bronchopneumonia. *Vet. Sci.* **2021**, *8*, 74.
2. Savitzky, A.; Golay, M.J.E. Smoothing and Differentiation of Data by Simplified Least Squares Procedures. *Analytical Chemistry* **1964**, *36*, 1627-1639.
3. McInnes, L.; Healy, J. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction, *ArXiv e-prints* **2018**, 1802.03426.
4. Löning, M.; Bagnall A.; Ganesh, S.; Kazakov V.; Lines J., Király F. sktime: A Unified Interface for Machine Learning with Time Series, 33rd Conference on Neural Information Processing Systems, Vancouver, Canada, 2019.
5. Faouzi, J.; Janati, H. pyts: A python package for time series classification. *Journal of Machine Learning Research* **2020**, *V. 21*, 1-6.
6. Schäfer, P. Scalable time series classification. *Data Mining Knowledge Discovery* **2016**, *30*, 1273-1298.
7. Minnaar, A. Time Series Classification and Clustering with Python, April 16, 2014. Available online: https://alexminnaar.com/2014/04/16/Time-Series-Classification-and-Clustering-with-Python.html. (accessed on 7 May 2021).