Voronezh State University of Engineering Technologies

High School of Economics

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

This research was supported by Russian Science Foundation, grant number 18-76-10015

Tatiana Kuchmenko, Anastasiia Shuba, Ruslan Umarkhanov, Vladimir Krylov

CSAC2021: 1st International Electronic Conference on Chemical Sensors and Analytical Chemistry
1-15 July 2021

Objective of the project

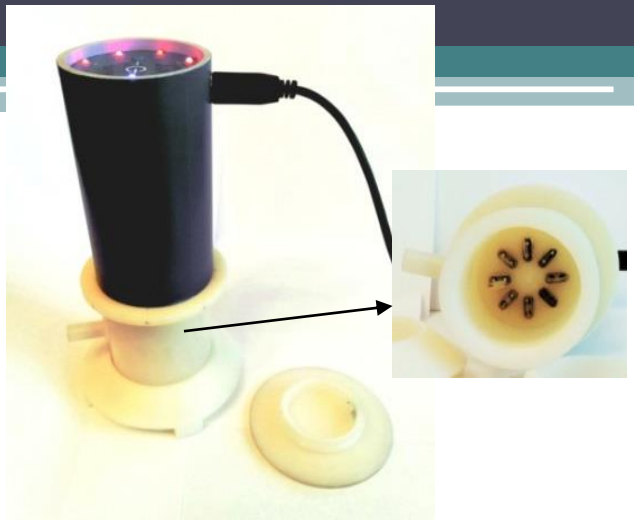
Here we will look at the development of a pipeline for stepwise processing of data coming from the sensors of the electronic nose. Our goal is to obtain a chain of software modules, the sequential application of which to raw data from sensors at the input generates the output number as the class that is most plausible for this data. Therefore, we have the classic classification problem. For the solution, we will construct a trainable model using the supervised learning method.

Methods and objects

The data from portable e-nose with eight piezoelectric sensors covered by nanostructured coatings 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)).



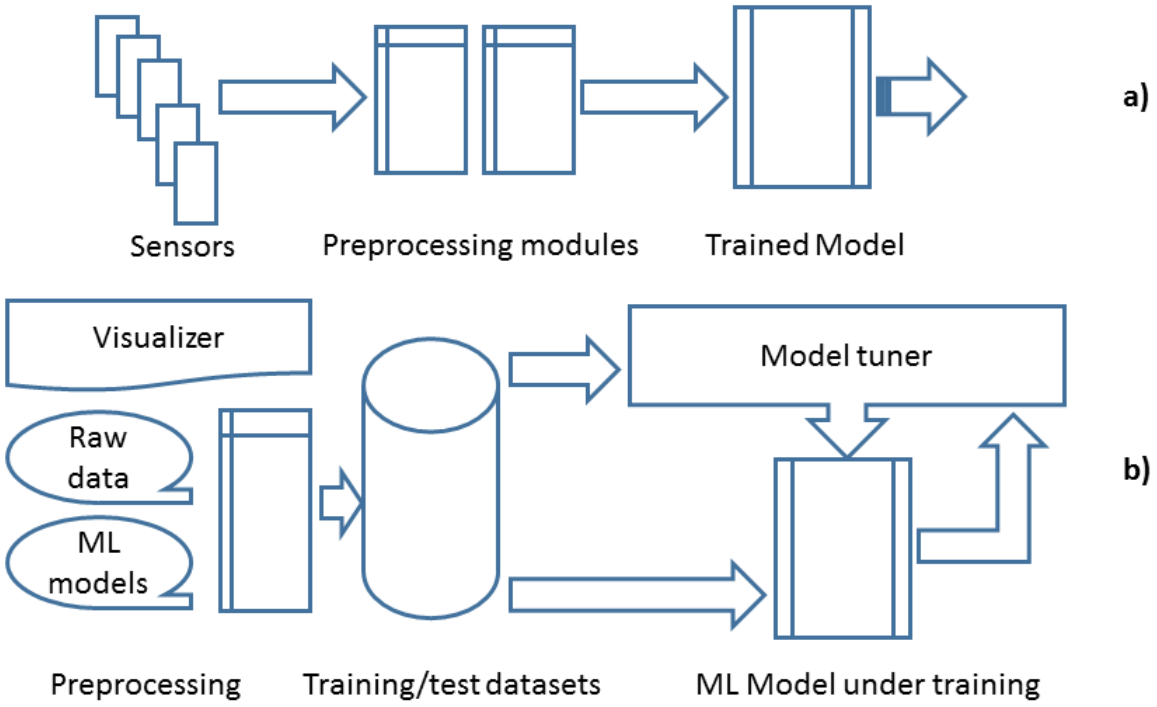
Modifier of piezoelectric quartz resonator electrodes:

Sensors 1, 8 – carboxylated carbon nanotubes of different mass (1-4 mcg), Sensors 2, 7 – phases of nitrate of zirconium oxide of different mass (2-4 mcg),

Sensor 3 – Dicyclohexano-18-Crown-6, Sensors 4, 5 – biohydroxyapatite phases of different mass (2-4 mcg),

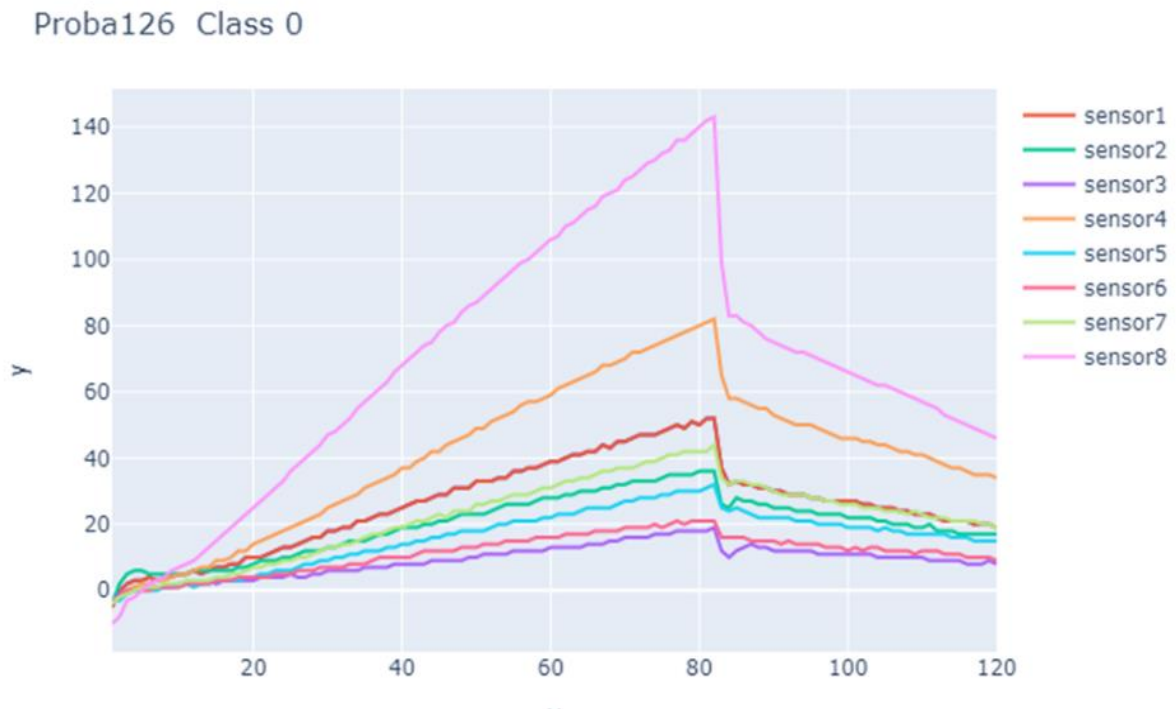
Sensor 6 – polyethylen glycol succinate.

Developing a processing pipeline in training mode



The result is the processing pipeline in inference mode shown in Figure (a). The main requirements for this pipeline are computational compactness and speed of classification, since focused on integration into a separate diagnostic device. 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 (b) shows this pipeline in training mode. The main requirement for this mode is the convenience of replacing models and assessing the model quality.

Chrono-Frequency-Gram's from the eight e-nose sensors for an experiment #126



According to the measurement mode of samples by e-nose, 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.

Sensors sequences derivative plots for sample #126

The sequences of values of the signals derivation from the sensors will be further used as input data. Their dimension is also 60 and they should be considered exactly as time series, but not just vectors of the same dimension. This is due to the specifics of the series. Their values do not mean as 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.

2-D data representation using UMAP technology

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 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 and pyts - A Python Package for Time Series Classification.

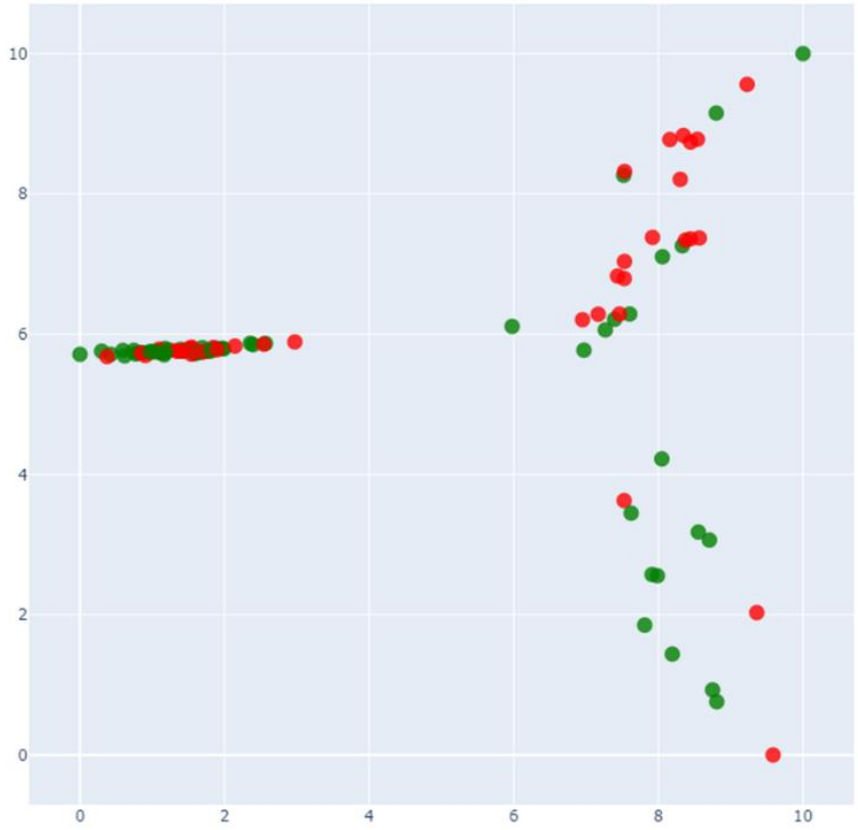


Figure shows labeled sick (red) and normal (green) data points.

Machine learning model development

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
Tree ensemble method, referred to as time series forest	sktime.TimeSeriesForestClassifier	0.6
Bag-Of-Symbols Ensemble technic	BOSSEnsemble	0.6
Univariate time series classifier which train linear classification models (logistic regression) with features extracted from multiple symbolic representations of time series (SAX, SFA).	MrSEQLClassifier	0.6
k-NN DTW Similarity	1-NNDTWClassifier	0.7

This machine learning model is a simple k-NN classifier, that is, 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 determination the distance between time series, we used a special metric for such series called dynamic time warping (DTW).

A fragment of the main module of the classifier model based on the use of the lower bound LB Keogh

```
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)
```

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.

Quality of obtained classification model on “sick”/ ”normal” classes for 75 experiments

	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.

Conclusions

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. The use of a machine learning model with the use of a special time series comparison metric made it possible to train the model to an accuracy of 83% even on 75 experiments.

Thank you for your attention

Contacts of authors:

Tatiana Kuchmenko, Tak1907@mail.ru, +7-920-422-7725
Anastasiia Shuba, Shuba1nastya@gmail.com
Ruslan Umarkhanov, rus_270487@mail.ru
Vladimir Krylov, vkrylov@hse.ru