MSc Thesis Computer Science

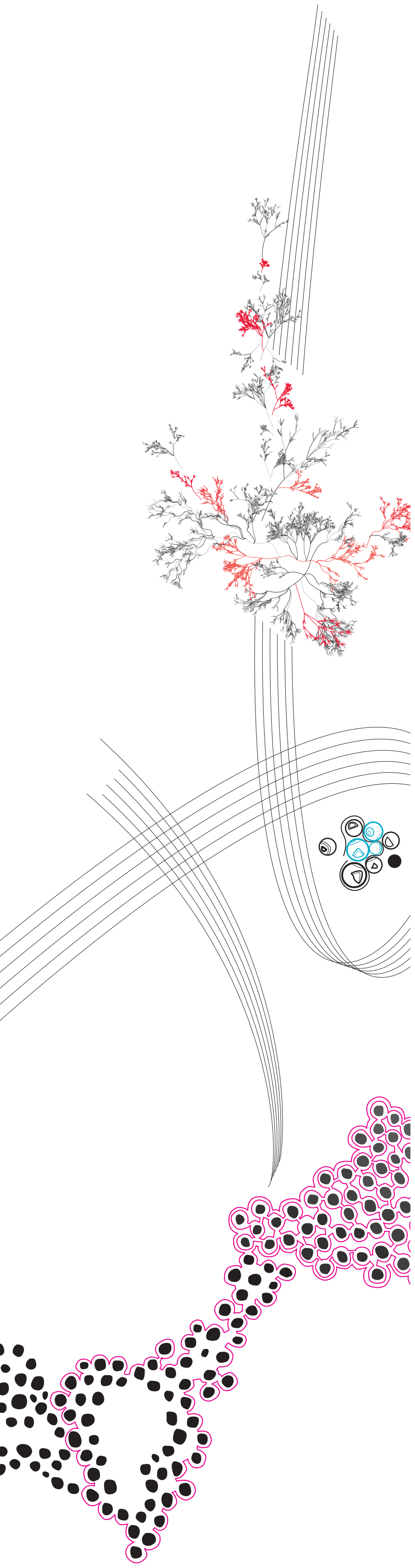
Supervised Contrastive learning to overcome inconsistencies in exhaled breath data

Ruben Henricus Lucas

Internal Supervisors:
dr. N. Strisciuglio,
dr.ing. F.W. Hahn

External Supervisors:
ir. H. Oord,
dr.ir. J.W. Gerritsen

August 28, 2022



UNIVERSITY OF TWENTE.

the eNose company

Department of Data Management & Biometrics
Faculty of Electrical Engineering, Mathematics
and Computer Science

Acknowledgements

My friend and fellow student Bas van Tintelen got me in contact with The eNose Company in Zutphen where I could conduct my master thesis in the highly innovative field of breath analysis with a challenging data problem. I want to thank Bas for introducing me to the company and especially for the great support throughout the whole process as we were both conducting our separate master theses at The eNose Company.

From the start till the end of my master thesis have I been supported by everyone working at The eNose Company. I want to thank them all for the constant feedback and support while also giving me the freedom to experiment with my own solutions and create my own planning.

I also want to thank my supervisor Nicola for providing me with new insightful ways to tackle encountered problems and for giving feedback on my work during our biweekly meetings.

Additionally, I would like to thank Florian Hahn for being part of my graduation committee and providing feedback on my work.

Supervised Contrastive learning to overcome inconsistencies in exhaled breath data

Ruben. H. Lucas
r.h.lucas@student.utwente.nl

August 28, 2022

Disease prediction can be performed based on breath analysis through the recognition of Volatile Organic Compound patterns. Such data is acquired through clinical studies where patients breathe into an electronic nose. This gives temporal data for every patient. Making disease predictions based on this data is challenging as the data is sparse, temporally complex, possibly non-linear, and contains inconsistencies. Supervised Contrastive learning tackles the device- and/or location-based inconsistencies that reside within the data by learning general features of the data through the recognition of similarities and differences between data points. This research shows that Supervised Contrastive learning learns more effective data representations to be used for disease classification on never before seen devices and thereby overcoming the inconsistencies. In doing so a more generalized classifier has been realized that can be used across multiple devices and locations.

***Keywords*: Exhaled breath analysis, Machine Learning, Supervised Contrastive learning, Temporal data**

1 Introduction

The potential of breath analysis has been recognized for a long time to be able to achieve health management, disease monitoring, and early disease diagnoses [1]. To perform breath analysis different Volatile Organic Compounds (VOC) have been identified. VOCs are derived from emitted gases that include a variety of chemicals and are also present in human breath. These VOCs serve as biomarkers [2] that indicate an ability to predict a future treatment response. These VOCs can be measured using a so-called electronic nose (e-nose) [3].

The eNose Company has developed their own e-nose called aeoNose with which they diagnose several forms of cancer and other diseases, based on VOCs from a person's breath, with the use of Machine Learning (ML) techniques. A schematic view of the aeoNose can be seen in figure 1.

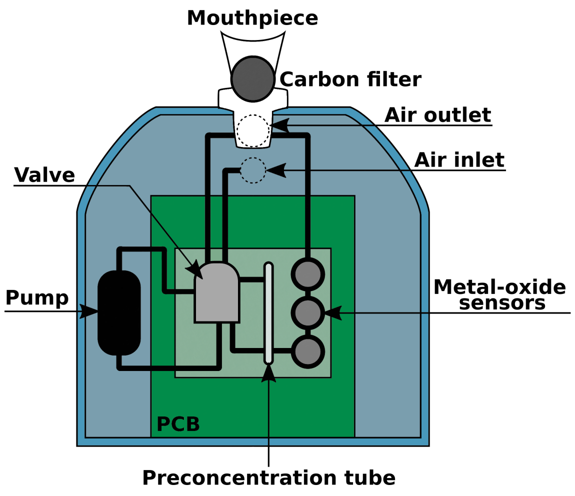


Figure 1: aeoNose schematic [4]

A single measurement with the aeoNose device is done by first flushing the device for 120 seconds, where the patient inhales air through the mouthpiece which is filtered by the carbon filter to cleanse the impurities from the device and the patient's lungs. Exhaled breath is directly outputted by the device and is not used for measurements at this moment. After the flush, a sample is taken in which a person's exhaled breath passes the three differently tuned metal-oxide sensors and exhaled air molecules are stored in the preconcentration tube. This sampling step takes another 180 seconds. After the sampling step, the patient can stop breathing through the aeoNose and the air present within the device is circulated for 240 seconds which we call the recovery period. Next, the preconcentration tube is heated for 30 seconds after which these preconcentrated VOC's are released upon the metal-oxide sensors for the last measurement of 190 seconds. The complete timeline of a single measurement can be seen in fig-

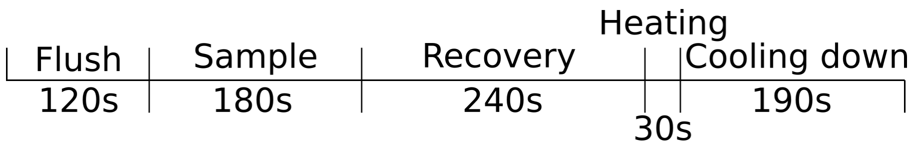


Figure 2: Measurement timeline [4]

ure 2. The metal-oxide sensors allow redox reactions to occur during the measurement in which the sensors are constantly heated and cooled between 260 °C and 340 °C according to two alternating sinusoidal waves [5]. This heating pattern is separately done from the earlier discussed 30 second heating phase of the pre-concentration tube. The first, longer sinusoidal wave has a frequency of 0.075 Hz and the second, shorter sinusoidal wave has a frequency of 0.15 Hz. This complete temporal sequence of conductivity measurements forms a single data point upon which a disease prediction is based.

Temporal breath data with an underlying disease can only be gathered by conducting clinical studies with the aeoNose device in the healthcare sector. This makes the gathering of data a difficult and time-consuming process. Moreover, the percentage of people that do have the disease that is searched for during the clinical study is significantly lower than the people that do not have the looked for disease. This results in an unbalanced data set with a lot of negative data samples and only a few positive ones. So, there is only limited unbalanced data available during this research.

Despite the carbon filtration and transferable metal-oxide sensor calibration models [6] that were developed by The eNose Company and built into the hardware of the aeoNose device, disease predictions are not always consistent when recorded with different device copies and when recorded in different locations. This means that there can be device- and/or location-based differences in the data. The exact cause of these differences is unclear, but is, according to The eNose Company, most likely caused by small manufacturing differences between the metal-oxide sensors of different devices [6] and/or because of differences in the VOCs contained within the ambient air of a location and/or differences in patient population.

Ideally, a prediction model should generalize to be able to predict the disease independent of external circumstances like device- and location-based differences. Most general solutions achieve a robust model by using large amounts of data to be able to distinguish between noisy features and more robust features that coincide within the data. Because of the small unbalanced data set, this is not an option for us. So, new solutions need to be investigated that can overcome these device- and/or location-based differences during the training of the disease prediction model while also dealing with the constraints of the data set.

To find an optimal way to improve the current classification performance and robustness with the constraints that reside within this given domain the following research questions have been set up.

RQ1 How can the current data pipeline of The eNose Company be improved?

- **SRQ1.1** How does the current data pipeline function?
- **SRQ1.2** What shortcomings can be identified in the current data pipeline?
- **SRQ1.3** What solutions could net the biggest improvement to the disease classification?

RQ2 What is the impact of Supervised Contrastive learning on the classification performance?

- **SRQ2.1** To what level do device- and/or location-based differences reside within the data?

This paper will answer these research questions by first explaining the preliminary knowledge required for this research in section 2. Next, in section 3 the data processing pipeline used by The eNose Company is analyzed to find shortcomings. In section 4 state-of-the-art solutions are explored that tackle the constraints of the data set and found shortcomings of the current data processing pipeline. A methodological approach to implementing the selected state-of-the-art solution in our current environment is set up in section 5. Preliminary results which can be found in section 6 and final results which can be found in section 7 are given based on the implementation of the methodological approach. Finally, in section 8 the research questions will be answered according to our findings.

2 Background

This section provides preliminary knowledge to clarify specific topics discussed in this paper and form a basic understanding of this research.

2.1 Synthetic Minority Over-Sampling Technique

Synthetic Minority Over-Sampling Technique (SMOTE) is a method to oversample a data set and solve the imbalanced data set problem. SMOTE is performed by taking the feature vector of minority class sample **c** and calculating the difference between it and the feature vector of one of its nearest neighbors **n** from the same class, retrieved using the K-nearest neighbors algorithm. This difference is multiplied by a random number r between 0 and 1, which is then added to the sample feature vector **c** to create a new feature vector **s**. So, a new feature vector **s** is created according to the following formula:

$$\mathbf{s} = ((\mathbf{c} - \mathbf{n}) * r) + \mathbf{c} \tag{1}$$

Hence the new feature vector **s** will lay along the line segment between the two neighboring feature vectors **c** and **n**. These steps can be repeated endlessly but are often repeated until the minority class is no longer underrepresented and has as many samples as the overrepresented class. With this method, new synthetic feature vectors are created that effectively force the decision region of the minority class to become more general [7, 8].

2.2 Singular Value Decomposition

One of the most used feature extraction methods in modern numerical analysis is the Singular Value Decomposition (SVD). The SVD can be applied to square [9, p. 78] and rectangular matrices [10] in the real and complex [11] domains by rearranging and merging information that resides within the matrices to create a matrix ordered by importance. This enables feature selection by specifying the desired number of features from the matrix starting from the most important one. The SVD can be used in this manner to reduce the number of linearly separable features of a data set while maintaining the majority of the information within the data.

The SVD rearranges and merges information for square matrices (rectangular matrices are given in square brackets) with the following theorem.

Let the data set be matrix $\mathbf{X} \in \mathbb{R}^{m \times n}$ consisting of m observations described by n variables and each measurement be represented by k_{mn}. Then there exists a square [rectangular] matrix $\mathbf{U} \in \mathbb{R}^{m \times m}[\mathbb{R}^{m \times k}]$, a rectangular [square] zero matrix with non-negative values on the diagonal $\mathbf{\Sigma} \in \mathbb{R}^{m \times n}[\mathbb{R}^{k \times k}]$, and a square [rectangular] matrix $\mathbf{V} \in \mathbb{R}^{n \times n}[\mathbb{R}^{k \times n}]$ [where $k = min(m, n)$] such that:

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \tag{2}$$

The columns of **U** and the columns of **V** are called left-singular vectors and right-singular vectors of **X**, respectively. For new data observations we get $\hat{\mathbf{X}} \in \mathbb{R}^{l \times n}$ where l is the number of data points in the new observations. Matrix $\hat{\mathbf{U}}$ is now determined by taking $\hat{\mathbf{U}} = \hat{\mathbf{X}}\mathbf{\Sigma}^{-1}(\mathbf{V}^T)^{-1}$. The number of columns respectively to the complexity and amount of information that is desired can be taken from $\hat{\mathbf{U}}$ to reduce the dimensional outcome compared to the original matrix **X** in which as much information is aggregated as possible [12, p. 166].

2.3 t-distributed Stochastic Neighbor Embedding

t-distributed Stochastic Neighbor Embedding (t-SNE) is a statistical method to visualize high-dimensional data similar to SVD but better optimized for visualization. Moreover, t-SNE is a nonlinear dimensionality reduction method and is, therefore, able to separate data that cannot be separated by any straight line.

t-SNE is implemented by determining the Euclidean distances between data points and converting them into conditional probabilities that are represented as similarities. In this way, the similarity of data point x_j to data point x_i is the conditional probability $p_{j|i}$, which conveys the probability that x_i would pick x_j as its neighbor. The Gaussian distribution is used as the probability distribution for $p_{j|i}$ where a perplexity value is given to determine the μ and σ. This gives us the following formula for the high-dimensional space p_{ij} where p_{ii} is set to zero.

$$p_{ij} = \frac{e^{\frac{-||x_i - x_j||^2}{2\sigma_i^2}}}{\sum_{k \neq l} e^{\frac{-||x_k - x_l||^2}{2\sigma_i^2}}} \quad (3)$$

A low-dimensional space q_{ij} with the same number of points is created by using a Student t-distribution with a single degree of freedom. Using this distribution helps to overcome the crowding problem where nearby and moderately distant data points are not accurately spaced out. This gives us the following formula for the low-dimensional map q_{ij} where q_{ii} is set to zero.

$$q_{ij} = \frac{(1 + ||y_i - y_j||^2)^{-1}}{\sum_{k \neq l}(1 + ||y_k - y_l||^2)^{-1}} \quad (4)$$

For the optimization of the mapping from the high-dimensional space p_{ij} to the low high-dimensional space q_{ij} gradient descent is used where the Kullback-Leibler divergence is minimized. This gives us the following cost function C.

$$C = D_{KL}(P||Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \quad (5)$$

The gradient of the cost function C with respect to y_i is given by the following formula.

$$\frac{\partial C}{\partial y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j)(1 + ||y_i - y_j||^2)^{-1} \quad (6)$$

By optimizing the cost function the low-dimensional space q_{ij} should represent the high-dimensional space p_{ij} as closely as possible. The low-dimensional space can now be used to understand the high-dimensional data. As t-SNE is not deterministic and iterative it is less useful for usage with ML algorithms as it produces different results for each run [13].

3 Current data pipeline

To go from raw exhaled-breath measurement to disease prediction a data processing pipeline has been setup by The eNose Company. The complete pipeline can be seen in figure 3.

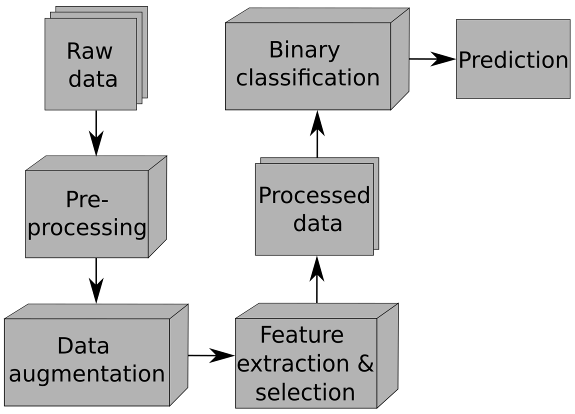


Figure 3: Current data processing pipeline of The eNose Company

3.1 Raw data

In this research a data set is used that was gathered during a clinical study for lung cancer. This is the only data set discussed and used throughout the whole report to limit the scope of this research. The data set consists of training data, which is split into a train and test set used for creating a classifier, and validation data, which is made up of new never before seen data points all from new never before used aeoNose devices. All of the devices present within the data set were also used in unique locations for different timespans. Determining if the earlier discussed device and/or location-based differences are caused solely by the different devices or locations is impossible. Consequently, we have decided to visualize the device and/or location inconsistencies based on the devices but it could be caused by a different phenomenon. The complete data set distribution can be seen in table 1.

Set	Devices	Pos	Neg	Total
Training data	5	234	312	546
Validation data	3	60	51	111

Table 1: Data distribution

The raw data of the data set is gathered with three metal-oxide sensors located within the aeoNose device. During a 750 seconds measurement, as described in section 1, conductivity values are gathered. Since the patient has to breathe through the aeoNose device for a period of time the obtained data is temporally correlated. This means that subsequent measurement readings are related to previous measurement readings as they are observed directly after each other. The resulting raw data consists of a temporal sequence of 2048 conductivity readings which is presented as 32 cycles each containing 64 conductivity readings called measurement points. One cycle of 64 measurements contains two sinusoidal waves, a combination of a short and long sinusoidal wave, known as subcycles. These 32 cycles contain the dirty air data, containing both the VOCs within the person's breath and ambient air. Additionally, we have a cycle with clean air data, containing only the ambient air.

3.2 Preprocessing

There are a plethora of preprocessing steps that can be applied to the raw data to remove unwanted outliers and organize the data set to make it suitable for feature extraction. Experimentation is conducted to find the best combination of these preprocessing steps for each type of disease.

Peak shaving

The sensors that are used to collect the exhaled-breath data are susceptible to slight errors and might therefore output spikes that do not represent the actual exhaled breath. These sudden spikes in the data should not be present since the redox reactions at the surface of the sensors, caused by the VOCs within the exhaled breath, are compared to the spikes slow chemical reactions that should influence the conductivity readings of the sensors in a similar slow reacting manner and thereby follow the temperature profile of the measurement. The spikes could also be caused by incorrect transformation readings of the analog-to-digital converter (ADC). Regardless of the cause of the inaccuracies, peak shaving can be used to limit its impact. This is done by reducing the peaks to the average of the surrounding conductivity readings.

Normalize signal data

The sinusoidal wave that is used to heat the sensors does not represent any information about a person's exhaled breath and therefore it might be beneficial to remove this wave from the data. As mentioned in section 1, the exhaled breath could contain device- and/or location-based differences. To limit this impact on the models the clean air data could be removed from the exhaled-breath data. Normalization of the signal data can be done by subtracting the clean air cycle from the exhaled-breath cycles. Additionally, the Fourier Transform (FT) can be taken from the data after which the exhaled-breath cycles can be multiplied with the complex conjugate of the clean air cycle and the resulting data point can be transformed back again with the FT, which is known as the Cross-Correlation [14]. In this way, normalization of the signal data could help remove the ambient air from data points.

Rescaling

The data can be rescaled by applying the natural logarithmic function to the data. This may deliver beneficial disease prediction capabilities because the natural logarithmic function removes exponential relations within the data. Alternatively, the exponent can be applied to the data to emphasize exponential relationships within the data. Finally, the square or square root can be taken from the data.

Sensor selection

The three metal-oxide sensors are composed of distinct materials and each reacts differently to the VOCs in the exhaled breath. This results in significantly different conductivity readings. Selecting individual sensors, or a combination of sensors, might contain more relevant information about the exhaled breath.

3.3 Data augmentation

To tackle the challenge of training a classification algorithm with a sparse data set data augmentation could be applied. SMOTE is used as it both tackles the sparsity as well as the unbalancedness of the data set [15] as discussed in section 2. Adding these synthetic data points can act as a regularizer in preventing overfitting in classification algorithms and improve performance.

3.4 Feature extraction and selection

Currently, each data point has 32 cycles each containing 64 measurement points. This high dimensionality, in combination with the limited number of available observations, results in a data set that is too complex for most ML methods to train an accurate classifier on. This phenomenon is called the curse of dimensionality where, in order to obtain a well-generalized classifier, the amount of data needed often grows exponentially with the dimensionality of the data [16]. To reduce the dimensionality of the data feature extraction and selection methods can be applied to the data. SVD can be used to extract features from the sequence of 2048 points ($32\ cycles \times 64\ measurements$). By applying the SVD we get a newly created matrix with features ranked from most to least important based on how well they represent the information within the original data. From this matrix, a feature selection can be performed by shrinking the matrix to the first n features that still contain enough information about the original data set based on the explained variance. After testing The eNose Company reached the conclusion of selecting 17 features to comply with the limited size of clinical studies as well as retaining enough information for classification.

3.5 Classification

With the raw data preprocessed, and possibly more data generated, an ML algorithm is trained and used for classifying people as either diseased or healthy. The most prominent ML algorithms used by The eNose Company are Catboost [17], Gradient Tree Boosting [18], Random Forest [19], and Extra Trees [20].

3.6 Data pipeline analysis

By analyzing the current data processing pipeline of The eNose Company different findings have been made. Firstly, no data normalization or standardization methods are applied to the data. Measurements often slightly differ in the range of actual values that are recorded. This is especially the case as the metal-oxide sensors slightly degrade over time and put out lower conductivity readings than newer devices. Normalization can be used to create a common scale between the different measurements and complete data set.

Secondly, the feature extraction method, currently the SVD, is a linear compression method. Therefore it obtains the relevant linear patterns that exist within the data set. However, the SVD is not able to capture non-linear patterns. As the data set obtained with the aeoNose device has high temporal complexity with an unknown distribution it is unlikely that there are only linear patterns within the data. So, the non-linearly distributed data is not taken into account, while this data could contain important information about the disease.

Lastly, the current ML methods used by The eNose Company cover a broad spectrum of the available classifiers except for the Deep Learning (DL) subdomain. Since the data is highly dimensional and potentially non-linear, Neural Networks (NN) could be more suitable than several of the currently applied ML methods as they consist of multiple layers that can composite higher-dimensional patterns for constructing classifiers. A downside of using deeper NNs is that most of them need a lot of data for training, which is a problem with our sparse data set.

4 Related work

To improve the current data processing pipeline we should find a way to tackle the identified shortcomings of the data pipeline while also taking into account the data characteristics. One of the most used methods to tackle the problem of having a sparse data set to train your ML algorithm is Transfer Learning (TL). TL is focused on improving the performance of a target task, in our case disease classification, with a target domain, in our case the conductivity readings of the metal-oxide sensors, by transferring the knowledge contained in different but related source domains. In this way, the dependence on a large number of target-domain data can be reduced for the construction of a target learner. Practically speaking, this method is similar to people who already know how to play the violin might learn to play the piano faster than those who do not have prior musical training, since both the violin and the piano are musical instruments and may share some common knowledge [21]. TL is most often used in the image domain but is slowly being adopted in other domains. In [22] multiple experiments of TL have been conducted to solve time series classification problems. These experiments show that the selection of a source and target domain is of essence and for a large part determines the effectiveness of the implementation. Therefore finding another data set and classification task that closely resembles the desired classification objective is essential. Choosing a source domain that does not closely resemble the classification objective could even harm the performance of the classification on the target task, which is called negative transfer [23]. Finding a data set that is similar to the metal-oxide sensor readings is however difficult as the sensors are purpose-made for the aeoNose device. Moreover, public data sets on temporal air molecule-based signals are almost non-existing or of limited size due to the constraints of clinical studies.

Another way to create a classification baseline without the need for another data set is by augmenting the current data set to create a new arbitrary classification task. [24] has created a framework that uses Contrastive learning to learn a baseline in a self-supervised manner. Recently this framework has also been modified to work in a supervised manner called Supervised Contrastive (SupCon) learning [25]. As we mostly have labeled data using a supervised classification framework should be more appropriate. SupCon learning is used to learn the general features of a data set by merely recognizing the similarities and differences between data points. This is done by pulling together embeddings from the same class while pushing away embeddings from different classes. This principle is illustrated in figure 4, where positive samples from different devices are pulled together while pushing away negative samples from the same devices. This figure also shows how SupCon learning could help in overcoming the device differences by pulling together samples from the same class to form a more generalized agreement between all data samples of the same class. We have chosen to illustrate the device differences, but the same principle goes for any kind of location-based difference as well.

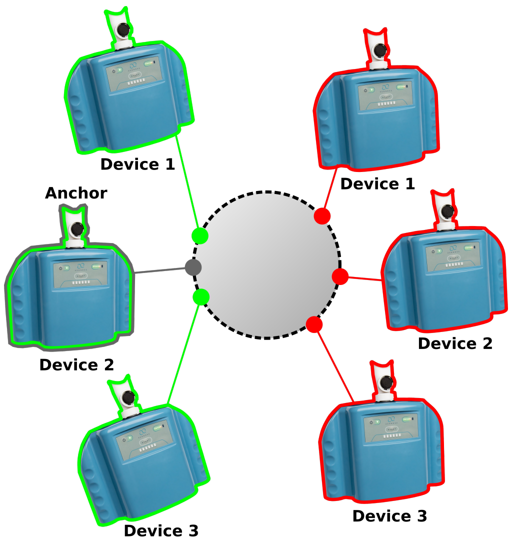


Figure 4: Supervised contrastive learning clustering (Adapted from [25])

SupCon learning is a new method from 2020 and therefore not that many implementations exist yet. [25] got state-of-the-art performances with the SupCon framework on many well-established image classification benchmarks. [26] used SupCon for kinship recognition where it aims to determine whether different images have a kin relation. This is done by pulling together facial images of family members and pushing away other facial images. Considering the classification of temporal data by using the SupCon framework we found an implementation for action recognition in videos [27]. For the data augmentation, this paper proposes to mix backgrounds from different videos which using the SupCon framework outperforms the addition of Gaussian noise to the video and two video mixup augmentations. Another implementation of the SupCon framework is in airwriting recognition [28]. Here temporal data recorded with an accelerometer and gyroscope sensor is translated into a letter of the alphabet that was written in free space using unrestricted finger movements. No data augmentations are described in this paper but results show that SupCon outperforms Cross-Entropy loss-based classifiers in this area.

5 Methodology

A methodological approach is established on how the SupCon framework can be effectively used with the temporal data of The eNose Company and where different structures and hyperparameters can be tested in a verifiable manner.

5.1 Proposed method

For our proposed method we decided to implement SupCon learning because it can help us overcome device- and/or location-based differences and does not rely on external data sets as discussed in section 4.

The SupCon framework works by first augmenting inputs. For every input data point two random augmentations are executed, $\tilde{x} = Aug(x)$, where each of the augmentations should reflect a subset of the information within the original data but illustrated as a different view of the data. More information on the type of augmentations is given in the next section.

An augmentation is then passed through an encoder network, which maps the augmented data point $\tilde{x}$ to a representation vector, $r = Enc(\tilde{x})$. This encoder network could consist of any type of NN but is most often build-up of convolutional layers.

The representation vector r is directly after the encoder network fed into a projection head which maps the representation vector to a projection vector, $z = Proj(r)$. The projection head should consist of a single or multiple dense layer(s).

This complete framework, $z = Proj(Enc(\tilde{x}))$, is trained with the usage of a SupCon loss function. Let $i \in I \equiv \{1, 2, ..., 2N\}$ be the index of an augmented sample from a batch of the training data set and let $P(i)$ be the set of indices of samples belonging to the same class as the i^{th} sample in the batch. Let $A(i)$ be the set of indices of samples in the batch other than the i^{th} sample. Now the SupCon loss function is defined as:

$$\mathcal{L}^{SupCon} = \sum_{i \in I} \frac{-1}{|P(i)|} \sum_{p \in P(i)} log \frac{e^{\frac{z_i \cdot z_p}{\tau}}}{\sum_{a \in A(i)} e^{\frac{z_i \cdot z_a}{\tau}}} \quad (7)$$

In the equation above, τ is a scalar temperature parameter and $\cdot$ represents the inner product of the surrounding variables. Minimizing the SupCon loss function encourages a generalized representation of the data by maximizing the agreement between differently augmented views of the same class. An illustration of the complete Supcon framework can be seen in figure 5.

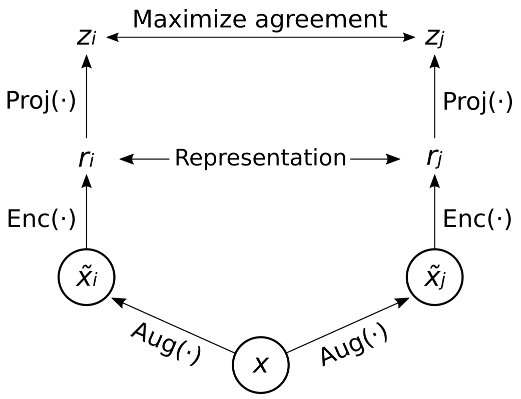


Figure 5: Supervised contrastive learning framework (Adapted from [24])

The gradient of the loss function with respect to z_i can be computed as:

$$\frac{\partial \mathcal{L}_i^{SupCon}}{\partial z_i} = \frac{1}{\tau} \left\{ \sum_{p \in P(i)} z_p \left(P_{ip} - \frac{1}{|P(i)|} \right) + \sum_{n \in N(i)} z_n P_{in} \right\} \quad (8)$$

In the equation above, $N(i)$ is the set of indices of samples belonging to the other class as the i^{th} sample in the batch and $P_{ix} \equiv e^{(z_i \cdot z_x/\tau)} / \sum_{a \in A(i)} e^{(z_i \cdot z_a/\tau)}$. Due to the structure of the SupCon loss harder positive and negative pairs have a larger contribution towards the gradient compared to easier positive and negative pairs providing an intrinsic mechanism for hard positive/negative mining during training [25, 28].

After the encoder and projection networks have been trained using the SupCon loss the projection head can be discarded. The projection head is exchanged for a classification head. This classification head should, just like the projection head, consist of a single or multiple dense layer(s), but untrained and without the SupCon loss. It should function as a normal classification network. As we have a binary classification problem that leads us to using binary cross-entropy loss for this new classification network. Moreover, the encoder network can also be locked to retain the current weights during the training of the classification network. Experiments should indicate if locking the encoder network completely, only some layers or no layers at all is most appropriate for our application.

5.2 Augmentations

As described in the previous section data augmentation is required for the implementation of SupCon learning. These augmentations should be appropriate for the temporal data with which we are working. The related work, discussed in section 4, didn't yield augmentations directly suitable for our data structure. Instead, [29] lays out multiple augmentations appropriate for temporal data. We have decided to use five of the laid-out augmentations, namely jitter, scaling, permutation, magnitude warping, and time warping. Jitter is performed by adding a random level of Gaussian noise to every time step. Scaling changes the global magnitude of a complete time series. Permutation rearranges segments of the time series in a different order to produce a new pattern. Magnitude warping multiplies the time series with a smooth curve created by interpolating a cubic spline. Time warping also uses cubic spline interpolation to create a smooth curve, but uses it to modify the time series in the temporal dimension. The effects of all the augmentations can also be seen in figure 6, where a single data point is augmented according to the five different augmentation methods.

5.3 Validation

Validation is essential as this research takes place in the health care sector where we are dealing with people's lives. Moreover, due to the nature of NNs it is nearly impossible to retrieve the exact reasoning as to why a NN has come to its prediction, which makes scientific reasoning about the decision-making process difficult. Therefore having a solid validation process is a way to demonstrate the robustness and performance of a NN model.

To validate the performance of an ML implementation we propose to use the signed-rank Wilcoxen test [30],

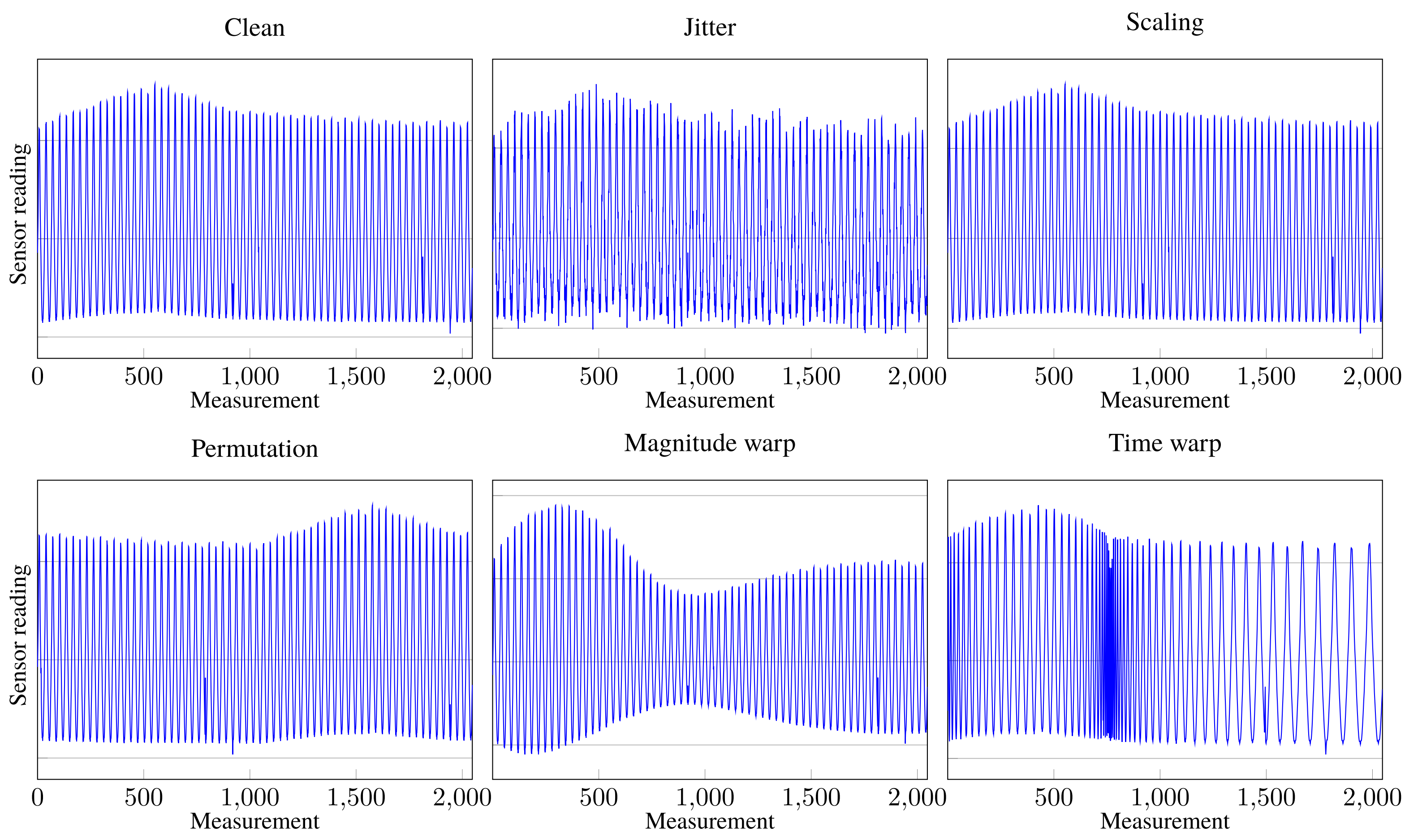


Figure 6: Augmentation overview on single data point

which is a non-parametric statistical hypothesis test on dependent samples. Non-parametric means that the test does not require the population to have an underlying distribution, which is preferred because of the high temporal complexity of the data and no known underlying distribution within the data. Moreover, the compared implementations will use the same samples to be trained upon which means that they are dependent.

For the implementation of the signed-rank Wilcoxen test, we propose to use the Area Under the Receiver Operating Characteristic (AUROC) as a performance measure. More specifically, the Area Under the Curve is computed from the Receiver Operating Characteristic (ROC) curve which itself is based upon the True Positive Rate (TPR) and the False Positive Rate (FPR). Therefore the ROC curve visualizes the trade-off between patients that have the disease while also getting predicted as having the disease against patients not having the disease while getting predicted as having the disease. So, AUROC tells us if the model is capable of distinguishing between classes [15, 31]. For the recording of the AUROC cross-validation has been used, which is a resampling method where different portions of the data are used to represent the train and test data for training the ML model. However, in this report only the aggregated results are shown consisting of the average test set and validation set AUROC results from 100 separate runs, where the test set is balanced between classes and does not include any synthetic data. Moreover, the test set is made up of 25% of the complete data set of which, for every run, this split is randomly taken. The validation set is an entirely new data set with no matching data points. The validation set is made up of new devices and test locations and should therefore give a good representation of the generalizability of the classifier, especially considering the device- and/or location-based differences that coincide within the data. This complete validation setup should ensure that the classification model that has the highest classification performance, and is consistently achieving this performance, on unseen data is chosen.

Extra trees classifier - Average of 100 runs				
Baseline #	Processing steps	# of cycles	Set	AUROC
0	Peak shaving - subtract - SVD	32	Test	59.29
			Val	48.96
1	Peak shaving - subtract - SVD	16	Test	62.55
			Val	51.05
2	Peak shaving - natural logarithm - SVD	16	Test	68.07
			Val	49.39

Table 2: Baseline results

6 Preliminary experimentation

Before the final results can be gathered with the SupCon framework, preliminary experiments were conducted to establish the best performing baselines to which the SupCon network is compared. Moreover, to establish how to implement the SupCon framework preliminary experimentation is performed on data processing steps, augmentations, and the NN architectural layout.

6.1 Cycle selection

Before training the classification model with the current data processing pipeline of The eNose Company discussed in section 3 and the SupCon framework discussed in section 5 we also decided to look at where in the data the most disease-indicating information resides. If most of the information resides in a subsection of the data it could be used to limit the number of time steps. This would limit the temporal complexity of the data which could help the training of an ML classifier. We have decided to look at the data in a cycle-based manner, so separately go through each of the 32 cycles and look at their impact. Such a cycle selection has the same goal as the feature extraction and selection discussed in section 3 but instead of transforming the data, the data is limited. In this way, the temporal structure of the data stays intact which could help the ML classifiers better understand temporal patterns.

To test in which cycle the most information resides we train the SupCon network discussed in section 5 on a standard training set consisting of all cycles. After training, we mask every cycle except the one we are currently investigating for every data point in the test set. We use the trained model to see the neuron activation output from the projection head and we look at the AUROC performance when predicting with the classification head. These steps are repeated for every cycle. We are looking at the neuron activation difference between positive and negative samples because these classes should be pushed apart by the SupCon framework and therefore a difference must become visible between both classes when the SupCon network finds a suitable representation. Furthermore, we look at the AUROC performance as this should give an indication of how much disease-indicating information resides within the cycle. An overview with results for every cycle can be found in Appendix A. From this experimentation, we have chosen to use cycles 10 to 25 as all the cycles in this range have a high difference between the positive and negative samples and a high AUROC compared to the other cycles. This cycle selection also lines up closely with the recovery period of a measurement conducted with the aeoNose device.

6.2 Baseline

According to the data processing pipeline of The eNose Company discussed in section 3 three different baselines have been set up. Baseline 0 reflects the results achievable at the beginning of this research. In baseline 1 the cycle selection is introduced, which is discussed in the last section. For baseline 2 the natural logarithm is applied to the data which showed more promising results than the processing steps of baseline 1. Different standardization and normalization steps improve slightly upon baseline 2, but these improvements are not significant and therefore not deemed necessary steps in the data preprocessing pipeline. An overview of the three baselines and their performances can be found in table 2. All baselines use the Extra Trees classifier [20] as this was overall the best performing classifier, but other ML algorithms can also reach comparable results. The baseline results on the test set are according to the Wilcoxen test significant improvements over the previous one. For the validation set which consists of new devices and locations, there are only slight differences and the performance still could be considered for all of the baselines as good as a random guess. All different data processing pipelines

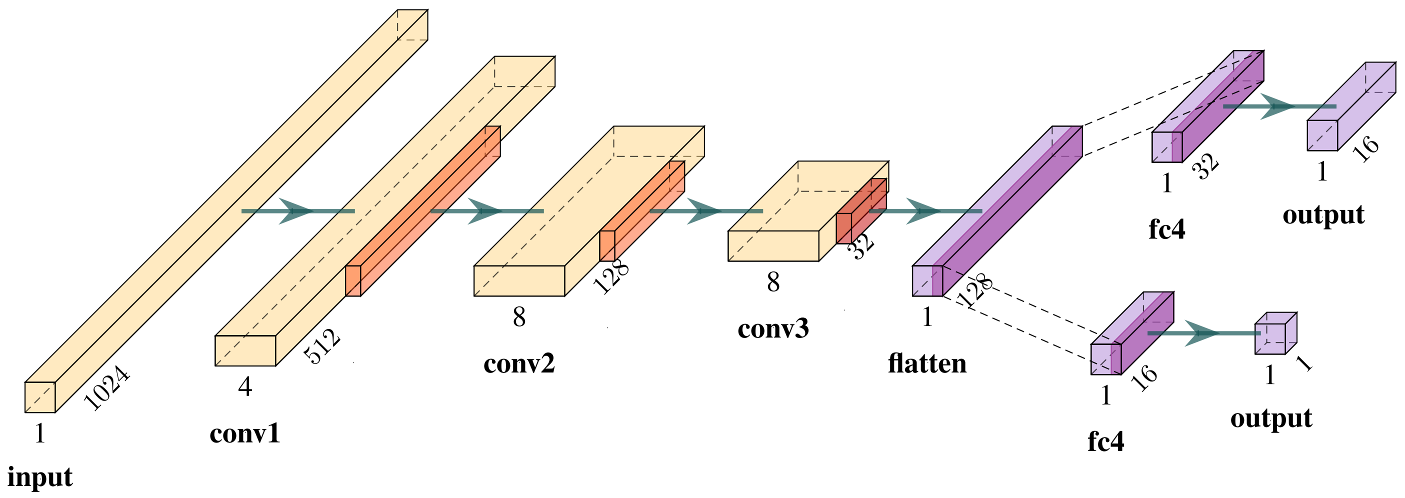


Figure 10: SupCon NN architecture [32]. At the left is the encoder network consisting of an input layer, three convolutional layers with two average pooling and one max pooling respectively, and a flatten layer. The projection head consisting of one fully connected and an output layer is layed out at the top right. This projection head is later on discarded and in its place is the classification head attached which is layed out at the bottom right. This classification head also consists of one fully connected and an output layer.

with their corresponding results that were tested can be found in Appendix B.

6.3 Experimental setup

For all experimentation with the SupCon framework, the NNs are trained for 250 epochs with a batch size of 32 because early testing showed that with these settings the network is always fully trained without considerable overfitting based on the test and validation losses.

The NN architecture of the SupCon framework can be seen in figure 10. The layout of this network has been established after rigorous testing with the validation strategy laid out in section 5.3. Architectural test results can be seen in Appendix C. During the testing of different NN architectures, the performance of the validation set is considered more important than the performance of the test set. Several regularization methods were also tested, but none of them netted any performance improvement. This was expected as we are dealing with a sparse data set in which overfitting is rather uncommon.

Experimentation with the augmentations discussed in section 5.2 showed that the jitter augmentation with a standard deviation of 0.1 gave the best results. Results on all different augmentations can be seen in Appendix D.

6.4 Device and/or location specifics

The data set used in this research consists of 8 different devices of which the distribution can be found in table 3. Devices one to five are used in the train and test sets, while the last three devices are used in the validation set.

Device	Positive	Negative	Total
1	81	73	154
2	20	33	53
3	39	29	68
4	6	54	60
5	12	47	59
6	26	27	53
7	19	11	30
8	15	13	28

Table 3: Device distribution

To make the device- and/or location-based differences visible we make use of t-SNE, as discussed in section 2, to visualize the high-dimensional data as a 2D plot. Figure 11 shows multiple t-SNE plots laying out the data visualized by the different device and target labels of all data points. The device plots A and B show clear clustering which indicates that there are differences between the separate devices. Moreover, target plots D and E show a random mix without any clear clustering which indicates the difficulty of the classification task. Ideally, the different devices should have no clustering and the targets should have clustering so as to have no device and/or location specifics in the data and making

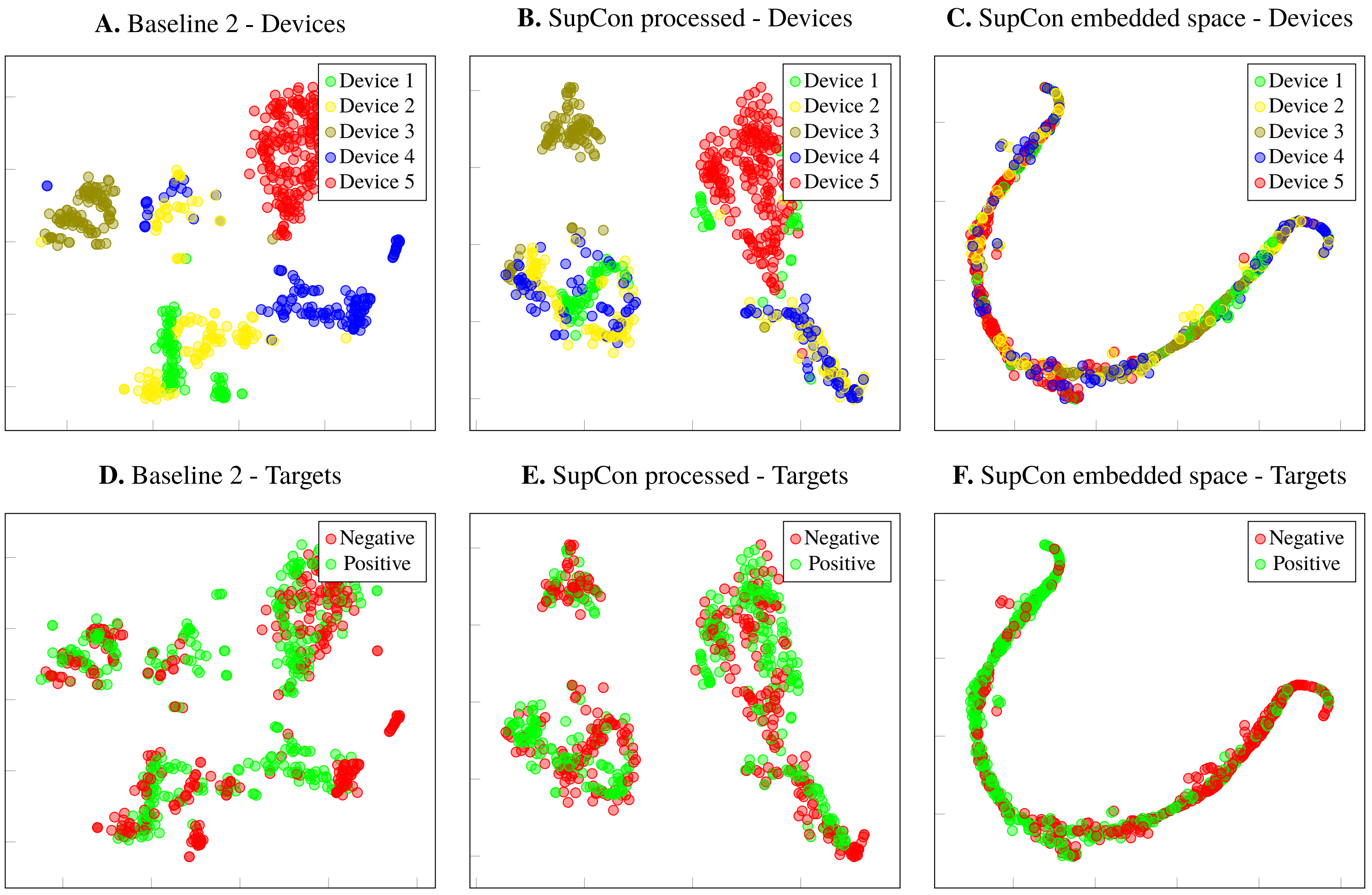


Figure 11: t-SNE plot comparison

the targets easily separable for a classification model.

Plots C and F show the embedded space acquired by putting all data points through the Supcon network with an encoder and projection head. The device clusters are mostly broken, indicating that the network should be less susceptible to device specifics and therefore we should be able to train a more generalized classifier. The clustering could however be improved further as some devices are still not evenly represented throughout the line of data points and while the negative and positive data points show a clearer separation we still don't see two clear clusters. More information on how the unseen devices of the validation set relate to the seen devices can be seen in Appendix E.

7 Results

By implementing SupCon learning, as discussed in section 5, we get the results shown in table 5. While the AUROC on the test set goes down 2.4 percent compared to baseline 2, the AUROC on the validation set goes up by 10.75 percent. Moreover, AUROC results with the SupCon framework can be reached that are higher than the test set performance of 68.07, as shown in Appendix C, but this would require a broader network that performs worse on the validation set. This is most likely caused by the network memorizing data characteristics and thereby overfitting on the data which leads to a less generalized classifier.

Average of 100 runs		
Device	Set	AUROC
1	Test	64.65
2	Test	77.00
3	Test	63.33
4	Test	74.20
5	Test	62.84
6	Val	68.66
7	Val	51.63
8	Val	66.21

Table 4: SupCon results per device

Average of 100 runs				
Network	Processing steps	Classifier	Set	AUROC
Baseline 2	Peak shaving - natural logarithm - SVD	Extra trees	Test	68.07
			Val	49.39
SupCon	Peak shaving - natural logarithm - standardization	NN (fig. 10)	Test	65.67
			Val	60.14

Table 5: SupCon results

An overview of the AUROC results per device is shown in table 4. This table shows that the SupCon network also achieves significantly higher results compared to random guessing across almost all devices. An interesting point to note is that a higher number of data points for a device does not directly result in a higher performance with the SupCon network. The exact reasoning for this occurrence is difficult to determine but a likely cause is that the classifier overfits on the large number of synthetic data points that are generated for these devices that do not have many real data points. Because we use SMOTE to generate synthetic data points, these synthetic data points are similar to the real data points on which they are based, hence pushing the classifier quite strongly into a single direction during training and making overfitting a likely outcome.

8 Conclusions

Based on the results of this study we can draw the conclusion that by implementing the SupCon framework proposed by [25] we have overcome the indicated problem of the device- and/or location-based differences within the temporal data. This was done on the basis of six research questions. More specifically, the research questions have yielded the following answers:

SRQ1.1 How does the current data pipeline function?

The current data pipeline of The eNose Company as in detail discussed in section 3 consists of multiple preprocessing options whereafter feature extraction and selection are performed with the help of SVD to limit the temporal complexity of the data. The modified temporal data is then used to train an ML algorithm of which the Extra Trees classifier is the most robust high performing algorithm currently used by The eNose Company.

SRQ1.2 What shortcomings can be identified in the current data pipeline?

By analyzing the data pipeline of The eNose Company we have identified multiple shortcomings. Firstly, we have identified that no normalization or standardization methods have been implemented which could help create a common scale between different data points. Secondly, SVD which is used as a feature extraction method is a linear compression method and can therefore not capture non-linear relationships that coincide within the data. Lastly, the DL subdomain is not explored much while NNs could capture higher-dimensional patterns for constructing classifiers that could fit well with the temporal complexity of the data.

SRQ1.3 What solutions could net the biggest improvement to the disease classification?

We have chosen to implement the SupCon framework proposed by [25] as this state-of-the-art solution tackles the characteristics of the data and the identified shortcomings. More specifically, SupCon learning uses two augmentations of every data point and because of that doubles the amount of data in our sparse data set. Device- and/or location-based differences are avoided by the SupCon network as it is actively breaking up unwanted clusters and forming clusters for the targets to make the classification task easier. Instead of the linear compression method, namely the SVD, an encoder is used by the SupCon architecture which could also capture non-linear relationships that reside within the data.

RQ1 How can the current data pipeline of The eNose Company be improved?

The current data pipeline of The eNose Company can be improved with the SupCon framework by using the jitter augmentation and a network combination of a convolutional encoder, a fully connected projection head, and a fully connected classification head as experimentation has shown.

SRQ2.1 To what level do device- and/or location-based differences reside within the data?

t-SNE plots shown in section 6.4 have demonstrated that clusters are formed according to specific aeoNose devices while the targets are randomly scattered and not clustered in any way.

RQ2 What is the impact of Supervised Contrastive learning on the classification performance?

The SupCon learning implementation broke the device clusters mostly and formed a slight separation of the target classes which was illustrated in section 6.4. This indicates that the learned classifier is able to recognize patterns within the temporal data that generalize across devices and locations. Moreover, the SupCon learning implementation achieved a performance increase on the validation set with unseen devices from 49.39 AUROC with the previously best performing baseline to a 60.14 AUROC. Altogether, the broken device clusters together with the performance achieved on never before seen devices proves that SupCon learning can effectively be used in a sparse temporal environment.

8.1 Limitations

Limitations were encountered because of the scope of this research. Starting with that all testing done during this research was performed on a single sparse training and validation data set combination of 505 data points. Moreover, this data set combination only covers a single disease, namely lung cancer, while the aeoNose device could be used for detecting multiple diseases.

Despite making the training set balanced by using SMOTE, there was chosen to not equalize the prevalence of data points across devices so that each device has an equal number of data points. Not equalizing the prevalences for devices was done as this would require an implementation that used SMOTE to create new samples for both the underrepresented and overrepresented target classes which would infer a lot of synthetic data. Alternatively, this would require us to delete data samples until every device had the same number of samples. Having devices represented equally might impact the performances of the SupCon learning implementation but will never nullify the impact that SupCon can have on creating a more generalized classifier that can overcome device- and/or location-based differences.

8.2 Future work

Future work needs to be performed using multiple new data sets on potentially new diseases to access the generalizability more broadly across the field of breath analysis. Internal testing at The eNose Company already indicated that changes to the processing pipeline and cycle selection assessed in section 6 might be needed as these diseases react differently with the metal-oxide sensors of the aeoNose devices.

Further exploration also needs to be conducted to find the underlying reasoning for the device and/or location inconsistencies. This is rather difficult with the current data set as it does not consist of a substantial number of devices used in a single location. This makes the decoupling of inconsistencies, occurring because of different devices and locations, impossible. A new data set consisting of multiple devices used at a single location in a strictly set timespan might shed light on these inconsistencies more clearly by visualizing them with a t-SNE plot.

Despite the doubling of the number of samples used to train a classifier in SupCon learning the sparsity of the data set is still considered one of the weaknesses present in the overall network. Further application of data augmentation to generate more data points is seen as one of the ways to tackle the sparsity while still making use of SupCon learning. This could also be used to directly tackle the limitation of not having equal prevalence across devices. Furthermore, data generation could be used to generate entirely new data points. A state-of-the-art application that could be investigated is data generation with a generative adversarial network [33]. Alternatively, TL could also be combined with SupCon learning. [34] has performed a study on TL with Contrastive learning. However, as discussed in section 4 a source domain that is closely related to our temporal data set needs to be found for this implementation to net a positive impact on the performance.

References

[1] H. Tai, S. Wang, Z. Duan, and Y. Jiang, "Evolution of breath analysis based on humidity and gas sensors: Potential and challenges," *Sensors and Actuators B: Chemical*, vol. 318, p. 128104, 2020.

[2] F. D. A. N. I. H. B. W. Group, *BEST (Biomarkers, EndpointS, and other Tools) Resource*. Food and

Drug Administration (US) and National Institutes of Health (US), 2016. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK326791/

[3] W. H. van Geffen, K. Lamote, A. Costantini, L. E. L. Hendriks, N. M. Rahman, T. G. Blum, and J. Van Meerbeeck, "The electronic nose: emerging biomarkers in lung cancer diagnostics," *Breathe*, vol. 15, no. 4, pp. e135–e141, 2019.

[4] B. F. M. van Tintelen and R. H. Lucas, "Schemetic and timeline created for master theses on behalf of the enose company," 2022.

[5] C. Baldini, L. Billeci, F. Sansone, R. Conte, C. Domenici, and A. Tonacci, "Electronic nose as a novel method for diagnosing cancer: A systematic review," *Biosensors*, vol. 10, no. 8, 2020. [Online]. Available: https://www.mdpi.com/2079-6374/10/8/84

[6] M. Bruins, J. Gerritsen, W. Sande, A. van Belkum, and A. Bos, "Enabling a transferable calibration model for metal-oxide type electronic noses," *Sensors and Actuators B: Chemical*, vol. 188, pp. 1187–1195, 11 2013.

[7] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority oversampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[8] S. C. Wong, A. Gatt, V. Stamatescu, and M. D. McDonnell, "Understanding data augmentation for classification: when to warp?" in *2016 international conference on digital image computing: techniques and applications (DICTA)*. IEEE, 2016, pp. 1–6.

[9] C. C. MacDuffee, *The theory of matrices*, ser. Ergebnisse der Mathematik und Ihrer Grenzgebiete. 1. Folge. Springer Berlin, Heidelberg, 1933.

[10] C. Eckart and G. Young, "A principal axis transformation for non-hermitian matrices," *Bulletin of the American Mathematical Society, vol. 45*, pp. 118–121, 1939.

[11] L. Autonne, "Sur les groupes linéaires, réels et orthogonaux." *Bulletin de la Société Mathématique de France, Tome 30*, pp. 121–134, 1902.

[12] V. Klema and A. Laub, "The singular value decomposition: Its computation and some applications," *IEEE Transactions on Automatic Control*, vol. 25, no. 2, pp. 164–176, 1980.

[13] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[14] D. Lyon, "The discrete fourier transform, part 6: Cross-correlation," *Journal of Object Technology*, vol. 9, no. 2, pp. 17–22, 2010.

[15] X. Guo, Y. Yin, C. Dong, G. Yang, and G. Zhou, "On the class imbalance problem," *Fourth International Conference on Natural Computation*, vol. 4, pp. 192–201, 10 2008.

[16] E. Keogh and A. Mueen, *Curse of Dimensionality*. Boston, MA: Springer US, 2017, pp. 314–315. [Online]. Available: https://doi.org/10.1007/978-1-4899-7687-1_192

[17] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "Catboost: unbiased boosting with categorical features," *Advances in neural information processing systems*, vol. 31, 2018.

[18] J. H. Friedman, "Stochastic gradient boosting," *Computational statistics & data analysis*, vol. 38, no. 4, pp. 367–378, 2002.

[19] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[20] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Machine learning*, vol. 63, no. 1, pp. 3–42, 2006.

[21] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2020.

[22] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Transfer learning for time series classification," in *2018 IEEE International Conference on Big Data (Big Data)*, 2018, pp. 1367–1376.

[23] Z. Wang, Z. Dai, B. Póczos, and J. Carbonell, "Characterizing and avoiding negative transfer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 293–11 302.

[24] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," 2020. [Online]. Available: https://arxiv.org/abs/2002.05709

[25] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan, "Supervised contrastive learning," 2020. [Online]. Available: https://arxiv.org/abs/2004.11362

[26] X. Zhang, M. XU, X. Zhou, and G. Guo, "Supervised contrastive learning for facial kinship recognition," in *2021 16th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2021)*, 2021, pp. 01–05.

[27] A. Sahoo, R. Shah, R. Panda, K. Saenko, and A. Das, "Contrast and mix: Temporal contrastive video domain adaptation with background mixing," in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34. Curran Associates, Inc., 2021, pp. 23 386–23 400. [Online]. Available: https://proceedings.neurips.cc/paper/2021/file/c47e93742387750baba2e238558fa12d-Paper.pdf

[28] A. Tripathi, A. K. Mondal, L. Kumar, and P. AP, "Sclair: Supervised contrastive learning for user and device independent airwriting recognition," *IEEE Sensors Letters*, vol. 6, no. 2, pp. 1–4, 2022.

[29] B. K. Iwana and S. Uchida, "An empirical survey of data augmentation for time series classification with neural networks," *PLOS ONE*, vol. 16, no. 7, pp. 1–32, 07 2021. [Online]. Available: https://doi.org/10.1371/journal.pone.0254841

[30] R. F. Woolson, "Wilcoxon signed-rank test," *Wiley encyclopedia of clinical trials*, pp. 1–3, 2007.

[31] A. P. Bradley, "The use of the area under the roc curve in the evaluation of machine learning algorithms," *Pattern recognition*, vol. 30, no. 7, pp. 1145–1159, 1997.

[32] H. Iqbal, "Harisiqbal88/plotneuralnet v1.0.0," Dec. 2018. [Online]. Available: https://doi.org/10.5281/zenodo.2526396

[33] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, "Generative adversarial networks: An overview," *IEEE signal processing magazine*, vol. 35, no. 1, pp. 53–65, 2018.

[34] A. Islam, C.-F. R. Chen, R. Panda, L. Karlinsky, R. Radke, and R. Feris, "A broad study on the transferability of visual representations with contrastive learning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 8845–8855.

A Cycle selection

SupCon NN architecture - Average of 10 runs					
Cycle #	Pos DP neuron activation	Neg DP neuron activation	Difference	AUROC	aeoNose timeline (fig. 2)
1	5562.96	5555.47	7.50	54.23	Sample
2	5568.03	5569.73	-1.7	43.58	
3	570.67	5575.35	-4.68	46.02	
4	5574.42	5579.78	-5.36	47.91	
5	5579.03	5583.18	-4.16	45.95	
6	5581.54	5585.19	-3.65	48.30	
7	5584.08	5586.81	-2.73	46.98	
8	5587.40	5588.28	-0.88	46.43	
9	5586.91	5586.57	0.33	46.37	
10	5578.50	5571.93	6.57	53.31	Recovery
11	5573.35	5554.16	19.19	52.81	
12	5565.55	5541.88	23.67	50.79	
13	5558.86	5535.86	23.00	52.99	
14	5556.07	5530.20	25.87	53.55	
15	5551.55	5524.72	26.83	55.07	
16	5549.78	5521.83	27.95	51.35	
17	5546.11	5519.59	26.52	55.34	
18	5544.51	5516.34	28.17	53.16	
19	5543.24	5515.66	27.58	48.92	
20	5542.20	5514.13	28.07	55.55	
21	5541.65	5513.51	28.14	54.83	Heating
22	5541.09	5513.87	27.21	52.90	Cooling down
23	5543.87	5516.99	26.89	55.04	
24	5544.44	5526.84	17.60	52.23	
25	5546.93	5533.82	13.11	54.52	
26	5547.77	5535.49	12.28	49.60	
27	5545.61	5534.40	11.21	52.22	
28	5544.64	5531.89	12.75	49.69	
29	5543.15	5529.38	13.77	45.95	
30	5544.86	5527.25	17.61	45.75	
31	5542.15	5525.41	16.74	48.33	
32	5541.45	5523.73	17.72	48.96	

Table 6: Cycle selection results

B Processing steps results

Extra trees classifier - Average of 100 runs on test set			
Network	Processing steps	# of cycles	AUROC
Baseline 0	Peak shaving - subtract - SVD	32	59.29
Baseline 1	Peak shaving - subtract - SVD	16	62.55
Test 2	Peak shaving - Cross Correlation - SVD	16	66.40
Test 3	Peak shaving - Cross Correlation subcycle - SVD	16	64.08
Test 4	Peak shaving - exponential - SVD	16	66.81
Test 5	Peak shaving - square - SVD	16	65.84
Test 6	Peak shaving - square root - SVD	16	67.26
Baseline 2	Peak shaving - natural logarithm - SVD	16	68.07
Test 8	Peak shaving - natural logarithm - SVD - Norm 0->1	16	67.52
Test 9	Peak shaving - natural logarithm - SVD - Norm -1->1	16	68.22
Test 10	Peak shaving - natural logarithm - SVD - Stand mean 0 stdev. 1	16	67.93
Test 11	Peak shaving - natural logarithm - Norm 0->1 - SVD	16	67.88
Test 12	Peak shaving - natural logarithm - Norm 0->1 - SVD - Norm 0->1	16	68.08
Test 13	Peak shaving - natural logarithm - Norm 0->1 - SVD - Norm -1->1	16	68.15
Test 14	Peak shaving - natural logarithm - Norm 0->1 - SVD - Stand mean 0 stdev. 1	16	68.03
Test 15	Peak shaving - natural logarithm - Norm -1->1 - SVD	16	68.47
Test 16	Peak shaving - natural logarithm - Norm -1->1 - SVD - Norm 0->1	16	67.85
Test 17	Peak shaving - natural logarithm - Norm -1->1 - SVD - Norm -1->1	16	68.11
Test 18	Peak shaving - natural logarithm - Norm -1->1 - SVD - Stand mean 0 stdev. 1	16	67.97
Test 19	Peak shaving - natural logarithm - Stand mean 0 stdev. 1 - SVD	16	68.33
Test 20	Peak shaving - natural logarithm - Stand mean 0 stdev. 1 - SVD - Norm 0->1	16	67.93
Test 21	Peak shaving - natural logarithm - Stand mean 0 stdev. 1 - SVD - Norm -1->1	16	67.65
Test 22	Peak shaving - natural logarithm - Stand mean 0 stdev. 1 - SVD - Stand mean 0 stdev. 1	16	67.91

Table 7: Results processing pipelines

C NN architecture results

Average of 100 runs								
Network structure							Set	AUROC
Encoder			Projection head		Classification head			
F:4 K:5 S:2 Average pooling	F:8 K:3 S:2 Average pooling	F:8 K:3 S:2 Average pooling	32	16	16	1	Test	64.58
							Val	58.71
F:4 K:5 S:2 Average pooling	F:8 K:3 S:2 Average pooling	F:8 K:3 S:2 Max pooling	32	16	16	1	Test	65.67
							Val	**60.14**
F:4 K:5 S:2 Average pooling	F:8 K:3 S:2 Max pooling	F:8 K:3 S:2 Max pooling	32	16	16	1	Test	63.14
							Val	58.67
F:4 K:5 S:2 Max pooling	F:8 K:3 S:2 Max pooling	F:8 K:3 S:2 Max pooling	32	16	16	1	Test	65.38
							Val	57.76
F:4 K:5 S:2 Average pooling	F:8 K:3 S:2 Average pooling		32	16	16	1	Test	64.76
							Val	51.62
F:4 K:5 S:2 Average pooling	F:8 K:3 S:2 Max pooling		32	16	16	1	Test	66.42
							Val	57.57
F:4 K:5 S:2 Max pooling	F:8 K:3 S:2 Max pooling		32	16	16	1	Test	64.87
							Val	52.40
F:8 K:5 S:2 Average pooling			32	16	32	1	Test	64.04
							Val	55.12
F:4 K:5 S:2 Average pooling	F:8 K:3 S:2 Average pooling	F:8 K:3 S:2 Max pooling	32	16	32	1	Test	**70.20**
							Val	56.78
F:4 K:5 S:2 Average pooling	F:8 K:3 S:2 Average pooling	F:8 K:3 S:2 Max pooling	32	16	64	1	Test	67.63
							Val	54.74

Table 8: SupCon NN architecture results. F = # of filters, K = 1D kernal size, and S = stride of the convolutional layer. The numbering in the projection head and classification head represents the output size of the corresponding fully connected layer.

D Augmentation results

SupCon NN architecture - Average of 100 runs			
Augmentation	Setting	Set	AUROC
Jitter	Stdev. 0.05	Test	64.49
		Val	56.79
Jitter	Stdev. 0.1	Test	65.67
		Val	60.14
Jitter	Stdev. 0.15	Test	63.34
		Val	58.41
Scaling	Stdev. 0.05	Test	62.30
		Val	56.23
Scaling	Stdev. 0.1	Test	62.15
		Val	56.52
Scaling	Stdev. 0.15	Test	62.10
		Val	56.85
Permutation	Max segments 5	Test	65.30
		Val	55.12
Permutation	Max segments 10	Test	64.93
		Val	55.59
Magnitude warping	Stdev. 0.2 & knots 4	Test	60.90
		Val	55.03
Magnitude warping	Stdev. 0.4 & knots 4	Test	65.42
		Val	59.40
Magnitude warping	Stdev. 0.2 & knots 8	Test	64.33
		Val	57.21
Magnitude warping	Stdev. 0.4 & knots 8	Test	60.53
		Val	54.47
Time warping	Stdev. 0.2 & knots 4	Test	65.36
		Val	52.95
Time warping	Stdev. 0.4 & knots 4	Test	65.54
		Val	56.99
Time warping	Stdev. 0.2 & knots 8	Test	59.37
		Val	49.15
Time warping	Stdev. 0.4 & knots 8	Test	65.79
		Val	54.66
Magnitude warping + Jitter	Stdev. 0.4 & knots 4 + Stdev. 0.1	Test	65.36
		Val	55.82
Time warping + Jitter	Stdev. 0.4 & knots 8 + Stdev. 0.1	Test	63.83
		Val	58.13
Permutation + Jitter	Max segments 5 + Stdev. 0.1	Test	66.14
		Val	50.74
Magnitude warping + Time warping	Stdev. 0.4 & knots 4 + Stdev. 0.4 & knots 8	Test	53.17
		Val	50.00

Table 9: Jitter augmentation results

E t-SNE plots validation set

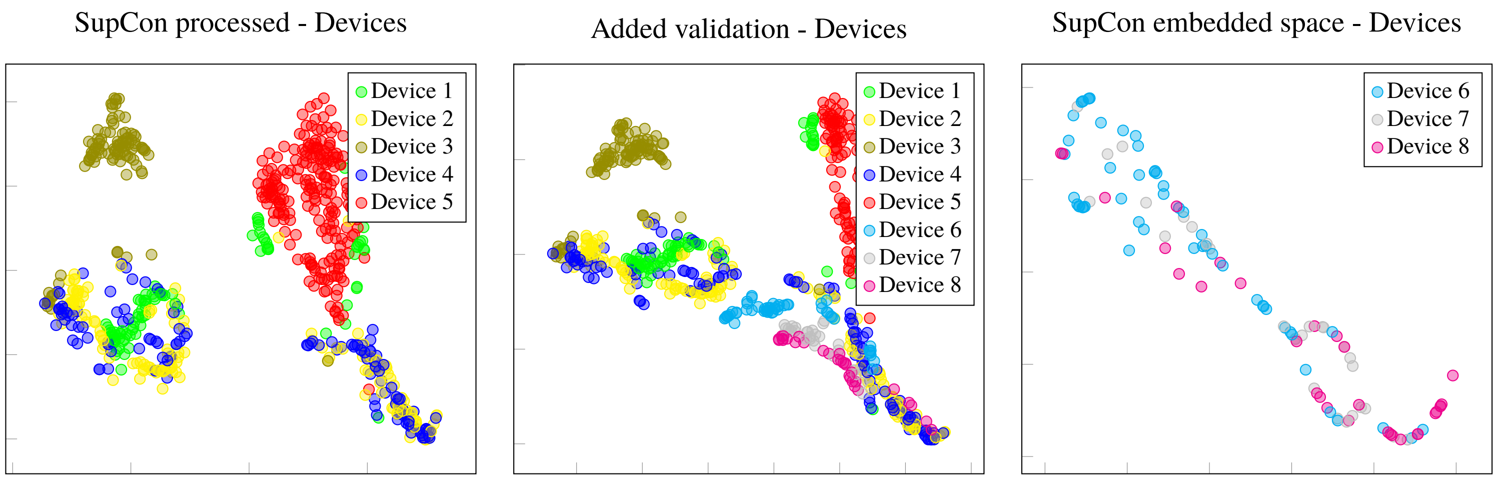


Figure 12: t-SNE plot comparison validation set on devices