

# Application of various balancing methods to DCNN regarding acoustic data

Dominic Schneider<sup>1</sup>, Manuel Schneider<sup>1</sup>, Maria Schweigel<sup>1</sup>,  
Andreas Wenzel<sup>1,2</sup>

<sup>1</sup>Embedded Diagnostic Systems, Faculty of Electrical Engineering,  
Schmalkalden University of Applied Sciences  
Blechhammer 9, D-98574 Schmalkalden, Germany  
E-Mail: d.schneider@hs-sm.de

<sup>2</sup>Fraunhofer IOSB, IOSB-AST Ilmenau, Fraunhofer Institute of Optronics,  
System Technologies and Image Exploitation  
Am Vogelherd 90, D-98693 Ilmenau, Germany  
E-Mail: andreas.wenzel@iosb-ast.fraunhofer.de

## Abstract

This paper describes the application and effects of different balancing methods on the learning behaviour and quality of a DCNN using acoustic data. The aim is to show to what extent these methods have positive as well as negative effects on the use case of the audio data. The evaluation is based on synthetic audio data with multiclass characteristics, because an overlay of effects should be avoided. This serves as preliminary work in order to apply the methodology to the measurement data for the classification of knife sharpness in forage harvesters in later investigations. According to applied balancing methods, the data are represented to the DCNN. The performance and quality shall be measured by formal qualification criteria. It turned out that SMOTE gives the best and most robust results. It shows a higher convergence compared to the other methods. Furthermore the worst results are produced with untreated raw data.

DOI: 10.58895/ksp/1000124139-4 erschienen in:

**Proceedings – 30. Workshop Computational Intelligence: Berlin, 26. - 27. November 2020**

DOI: 10.58895/ksp/1000124139 | <https://www.ksp.kit.edu/site/books/m/10.58895/ksp/1000124139/>

# 1 Introduction

The application of audio data as learning data set to neural networks can be subject to a wide range of use cases. The use case to be addressed in this paper is classification. This means, in supervised learning, that given class data is learned and subsequently the developed classifier can independently divide represented data into classes [1]. It also means that the aim is to assign the entered data to a class in the neural network. In real use cases there are usually problems with more than two classes. These are defined as multi-class problems. It can also happen that data sets acquired by measurements show unequally distributed class representations. These data sets are called unbalanced. The problem with these lies in the under-represented data, which can lead to a shift in recognition accuracy towards the over-represented data. There are now various methods for dealing with these unbalanced data. The influence of balancing methods has already been investigated on different data and learning structures.[2] They show that there is a probably existing influence on learning behaviour. Thus, the actual goal is to find an optimal balancing method for the real use case of the acoustic data of forage harvesters. This is also a classification problem and should be able to draw conclusions about the sharpness of the knives based on the structure-borne noise of the cutting drum. These audio data are measured values. In order to avoid an accumulation of effects in the evaluation, the effects will be investigated in this paper using synthetic data. This creates a basis which can be applied to the measured data in subsequent work.

The basis of the paper is synthetic data from a self-designed generator. This generator works on the principle of a recursive filter, more precisely the autocorrelation filter. The filter has the order  $N = 6$ . In total four classes are generated. For each class there is a corresponding representative, after which all class-related data is generated. The representative is audio data with a sampling rate  $f_a = 44.1 \text{ kHz}$ . The number of patterns is chosen very unbalanced so that the following data set is created:

The data for the following investigation are subject to the distribution according to table 1. Most of the articles deal with two-class problems, whereas this one already uses a four-class problem as in the following use case of forage

Table 1: Classification of the synthetic data set

class	sample
1	350
2	2500
3	4000
4	9500

harvesters. The uneven distribution is similar to the first real measurement series, with the sharpest knife condition being strongly underrepresented with class 1 and the duller knife condition being most with class 4.

## 2 Methods

### 2.1 Balancing Methods

The balancing procedures represent the methodological variation of the study. The conventional methods can be divided into five large groups.[3] The first group are the **Non-Heuristic Sampling Methods**. These work randomly and each delete or copy the data to arrive at a certain number of samples. These include Random Over-Sampling and Random Under-Sampling. Random Over-Sampling copies randomly selected data from the under-represented classes and adjusts them quantitatively to the most common class. Random Under-Sampling works the other way round and adjusts the data to the least occurring class. It randomly deletes data samples until the quantities are adjusted.[4]

The second group uses **Synthetic Data Generation**. These also approximate the data samples of all classes of a particular minority or majority. The Synthetic Minority Over-Sampling Technique (SMOTE) and the Adaptive Synthetic Sampling (ADASYN) are used for this. Both mentioned methods are over-sampling methods. The Synthetic Data Generation works in the feature space and less in the data space, like the Non-Heuristic Methods. The SMOTE algorithm over-samples the minority class by forming synthetic samples along

segments of the minority class of  $k$  nearest neighbors [5]. The nearest neighbors are determined randomly. Depending on the amount of over-sampling, a sample is generated in the direction of the determined nearest neighbor. Vectorially it is considered to form a difference vector. This is to be multiplied by a random number  $\delta = [0, 1]$  and appended to the original vector. In this way, a training record  $S$  can be considered with a minority class  $P$  and other classes  $N$ . For each sample  $\mathbf{p}_i \in P$  there is a  $k$  nearest neighbor  $\mathbf{x}$ . A new vector can now be defined as described above:

$$\mathbf{x}_{\text{new}} = \mathbf{p}_i + (\mathbf{x} - \mathbf{p}_i) \times \delta \quad (1)$$

This causes a selection of a point on a segment between a sample and its nearest neighbor.[6]

ADASYN is motivated by SMOTE and aims to reduce bias including adaptive learning. Looking at the training data set  $S$  with  $m$  samples,  $\{\mathbf{x}_i, y_i\}$ ,  $i = 1, \dots, m$ , we propose  $x$  as sample and  $y$  as label. Now  $m_s$  is described as the number of minority class samples and  $m_l$  as the number of majority class samples. Now the degree of imbalance is calculated:

$$d = m_s / m_l \quad (2)$$

Considering that  $d = (0, 1]$  now determines how many samples must be generated:

$$G = (m_l - m_s) \times \beta \quad (3)$$

The parameter  $\beta$  indicates the degree of balancing and is set to 1 in our case. Now, similar to SMOTE, we have to find the  $k$  nearest neighbour. This is created using the euclidian distance in  $n$  dimension and the radius is calculated:

$$r_i = \Delta_i / K, i = 1, \dots, m_s \quad (4)$$

Where  $\Delta_i$  describes the number of  $k$  nearest neighbours. Furthermore the radius has to be normalized in the following way:

$$\hat{r}_i = r_i / \sum_{i=1}^{m_s} r_i \quad (5)$$

Now it is calculated how many synthetic data samples must be generated per minority sample:

$$g_i = \hat{r}_i \times G \quad (6)$$

Where  $G$  is the absolute number of synthetic data examples. Now the following rule can be used to generate the corresponding synthetic samples for each  $\mathbf{x}_i$  sample:

$$\mathbf{s}_i = \mathbf{x}_i + (\mathbf{x}_{zi} - \mathbf{x}_i) \times \delta \quad (7)$$

The main difference to the SMOTE method is the use of a density distribution  $\sum_i \hat{r}_i = 1$  as a criterion at which point synthetic data must be generated.[3]

The third group uses **Cost-Sensitive Learning**. Here a cost factor for false predictions is determined, which affects the gradient. The data is not changed in this procedure. In this paper, the Weighting Factor is calculated using the samples per class. We assume there are  $N$  classes and the  $i$ th class has  $m_i$  training samples. Then a Classweighting  $w_i, i = 1, \dots, N$  can be determined, which will affect each class according to the number of its samples:

$$w_i = \frac{\sum_i m_i}{m_i \cdot N} \quad (8)$$

The last group represents the **Active Learning**. Here, learning algorithms are applied in advance to balance the data. One application is the methodology of Cluster-Centroids. Depending on the minority class, clusters are formed in the point clouds of the other classes. That means, assuming a learning data set  $N$  with  $m$  samples,  $k \geq 2$  subsets have to be found, which are represented by similar attributes. Thereby the single cluster focal points, but the centroids should differ from each other. With  $N = \{x_1, \dots, x_n\}$  a set of points is defined, which fulfill a similarity condition. It applies  $x_i \in \mathbb{R}^d$  and  $d \geq 1$ . Now  $\mu_j$  and  $M = \{\mu_1, \dots, \mu_k\}$  is defined as centroid of the cluster  $G(j)$  and a cost quantity  $P = \{G(1), \dots, G(k)\}$  is introduced. This describes the cluster problem and is used for optimization:

$$\mathbf{P}: \text{minimize } z(W, M) = \sum_{i=1}^n \sum_{j=1}^k w_{ij} d(x_i, \mu_j) \quad (9)$$

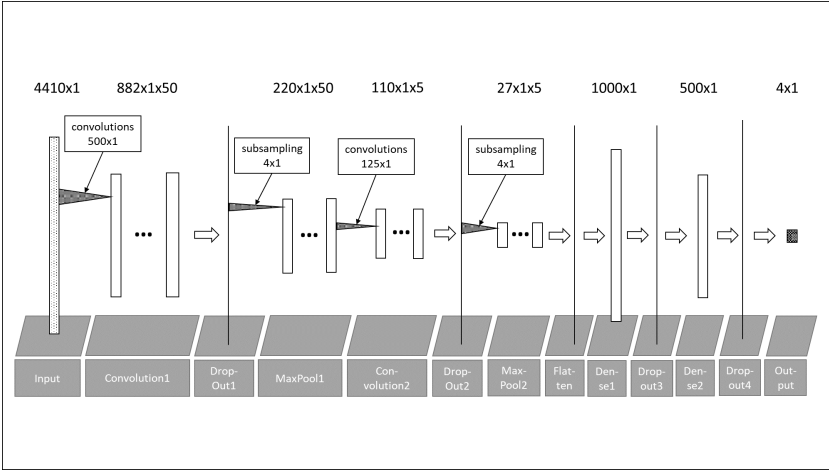


Figure 1: Netsummary of deep convolutional neural network

With  $w_{ij} \in \{0;1\}$  it is defined if  $x_i$  counts to  $G(j)$  and  $d(x_i, \mu_j)$  defines the euclidean distance. There are now several methods to solve  $\mathbf{P}$  which will not be discussed in detail. It should only be noted that  $k = m_s$  defines itself as a sample set of the minority class. This leads to the consequence that the formation of cluster centroids counts as an under-sampling procedure.[7]

## 2.2 Deep Convolutional Neural Network

The structure of a deep convolutional neural network was used for the investigation. This structure, as shown in figure 1, has not changed during the analysis. It was proceeded in such a way that there were several initializations, more precisely 100, of the network, because the weights  $w$  of the DCNN are initialized stochastically. This set  $I$ , in the following called individuals, is generated per balancing procedure and then evaluated. Thus the purely stochastic influence of the weight initialization crystallizes in the results.

The first layer is the **Input layer** of the net. This layer represents the audio samples of length  $n_s = 4410$ , thus  $t_s = 0.1\text{ s}$ . Furthermore, this is the first Convolution Layer. It is one-dimensional due to its application to audio vectors.

In the configuration, a filter depth of 50 windows has been set, with each filter window having a width of 500. The activation function is the rectified linear unit.

$$f(x) = \max(0, x) \quad (10)$$

The following **Dropout layer** in conjunction with the **Max- Pooling Layer** creates a stochastic selection of the activated output neurons of the convolution layer. Thus, it represents a multinomial distribution during the training period, with the effect of avoiding overfitting in the learning behavior. The dropout rate is 0.4 and the pooling rate is 4.[8] Following this, a second **Convolution Layer** is added. This layer has the same filter width of 500, but only 5 filter windows. The sense of a second convolution layer is the abstract filter behaviour. The first layer should find features on a lower level of abstraction and the second layer on a higher level of abstraction. Dropout and pooling layer are added to this layer to achieve the equivalent effect as with the first convolution layer. The dropout rate and pooling rate are the same as the previous ones with 0.2 and 4. The subsequent **Flatten Layer** is used for dimension reduction and is merely a preparation of the structure for the subsequent **Dense Layer**. These are completely connected layers and are characterized by precise learning behavior. The attached dropout layer also serves to prevent overfitting. The first dense layer has a size of 1000 units with an activation function of the hyperbolic tangent.

$$f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (11)$$

The following dropout layer has a rate of 0.5. The second dense layer is smaller and has only 500 units with an attached dropout layer and a rate of 0.5. The **Output layer** has the Softmax.

$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}} \quad (12)$$

It is one of the most frequently used components and is also applied here.[9] The **cost function** is the categorical crossentropy with applied optimizer ADAM.

$$\mathcal{L}_{CCE}(f(\mathbf{x}), y) = -\mathbf{e}_y \log f(\mathbf{x}) \quad (13)$$

## 2.3 Qualifying criteria

In the previous section the structure of the classifier was shown. The theoretical basics of the sampling methods used were also discussed in advance. The classification performance is of central importance for this paper, as it is the basis for the evaluation of the balancing methods. In most cases, scalar parameters are determined for this purpose, which have the static consideration of the correctly assigned classes. However, in most cases this does not yield meaningful evaluations. The problem may arise that in case of unevenly distributed classes, i.e. unbalanced data sets, the overrepresented classes are preferred. This way, less existing classes, which have the same valence, would hardly be recognized, but the classifier would still have a comparatively high classification quality. A further problem can arise if the relevance of the correct classification is very different. This is not the case with the data used, but can arise when applying the same methodology to real data. A third problem, which can occur with real data, is uncertainty or noise in the expert specifications. For all measurement data sets, which originate from real applications, the actual measurement signal is faulty. So an expert estimation of the target state of the data can also be faulty. Due to the synthetic data generation this will not be the case in this paper.

With the preceding illustrations of quality problems, the calculations of the used quality parameters shall now be shown. For this purpose, we first look at the feature data with  $N$  samples. The assignment of a class to a  $i$  data sample is described as  $CL_i$  by the classifier. The assignment by the expert is defined as  $CL_i^s$ . The element  $s_j$  of the vector  $\mathbf{s} = [s_1, \dots, s_N]^T$  is used to define an object of a test set of the class  $j$ . After calculation of the classification results an element  $r_j \in [0; s_j]$  of the vector  $\mathbf{r} = [r_1, \dots, r_N]^T$ , which represents the output of the classification. This vector contains the number of matching objects of a class  $j$ . This results in the simplest qualification criterion, which is called  $q_1$ :

$$q_1 = \frac{\sum_{j=1}^N r_j}{\sum_{j=1}^N s_j} \quad (14)$$

This includes the already mentioned problem of preferring the most common class. As a remedy, a criterion must be worked out, which also considers the

weakly occurring classes. For this purpose  $q_2$  is introduced as follows:

$$q_2 = \frac{1}{N} \sum_{j=1}^N \frac{r_j}{s_j} \quad (15)$$

By normalizing to the number of samples, all parts with the same weight enter the quality criterion. One way to calculate a measure for the worst recognized class is the geometric mean. This remains independent of the number of objects in a class and is defined as follows:

$$q_3 = \sqrt[N]{\prod_{j=1}^N \frac{r_j}{s_j}} \quad (16)$$

With the evaluation of the quotient  $r_j/s_j$  the class  $j$  can be examined for correct recognition. If the attention is now turned to the generally worst class after

$$q_4 = \min_{i=1,2,\dots,N} \left( \frac{r_j}{s_j} \right) \quad (17)$$

, the robustness of the classifier can be assessed. However, the parameter  $q_4$  is to be supplemented by a more robust measure. The classification can be regarded as a nominal scale, thus the kappa coefficient is to be used for the judgement agreement. Its starting point is the permutation matrix of two judges, the expert and the classifier. The form presented in the following contains the expert specification as rows of the matrix and the judgements calculated by the classifier as columns. The elements of the swap matrix  $\mathbf{K}$  are relative frequencies  $k_{ij}$ :

$$\mathbf{K} = \begin{pmatrix} k_{11} & \dots & k_{1N} \\ \vdots & \ddots & \vdots \\ k_{N1} & \dots & k_{NN} \end{pmatrix} \quad (18)$$

The frequencies of occurrence of target and actual classification are defined and counted as  $ka_{ij}$ . A normalization is done by the set of test data records  $T$  after  $ka_{ij}$ :

$$k_{ij} = \frac{ka_{ij}}{T} \quad (19)$$

The relative frequency of concordant judgments results along the main diagonal of the matrix  $K$ , as follows:

$$p_o = \sum_{i=1}^N k_{ii} \quad (20)$$

The proportion of expected concordant judgments is compared to this using the following relative frequency:

$$p_e = \sum_{i=1}^N \left( \sum_{j=1}^N k_{ij} \right)^2 \quad (21)$$

The above calculation is different from the original definition and uses the squares of the row sums instead of the products of the row and column sums. This is done to achieve an increased weighting of the target classification. The reason for this is that the original kappa coefficient assumes two independent and equal viewers, which is not the case when classifying with a neural network. The kappa coefficient itself is defined as follows:

$$\kappa = \frac{p_o - p_e}{1 - p_e} \quad (22)$$

Finally, there is also the demand for performance in the learning behavior of neural networks. For this purpose a parameter  $t_i$  is introduced, where  $i = 1, \dots, I$ . This parameter is measured during the calculation and specifies the calculation time for the learning process until a convergence criterion is reached. The background is that with over-sampling or under-sampling the amount of learning data is changed. This should be taken into account when considering the balancing methods. The quality, robustness and performance of the classifier is evaluated with the six criteria described above.[10]

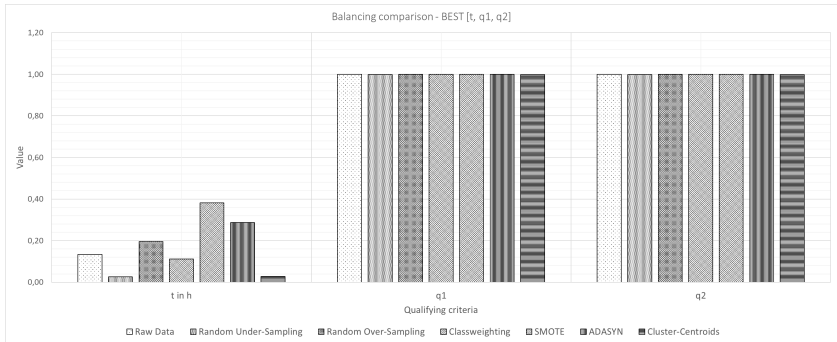


Figure 2: Balancing comparison of the best runs with  $[t, q_1, q_2]$

### 3 Results

As already mentioned, training was done with a total of  $I=100$  runs of the same data and the same balancing procedure. The only difference between these individuals lies in their stochastic initialization of their weights. At first the best trainings resultst are to mention. For this purpose the parameters  $t, q_1, q_2, q_3, q_4$ , and  $\kappa$  were used.

In figure 2 you can see that there are big differences in the area of training time. The SMOTE and ADASYN methods are the learning methods with the longest training times and the Random Under-Sampling with the Cluster-Centroids the shortest training times. This is due to the creation or removal of samples which actively influence the teaching time. The parameters  $q_1$  and  $q_2$  show that the maximum of 1 is reached as the best run and therefore there is no difference in the balancing procedures.

The analysis of the parameters  $q_3, q_4$  and  $\kappa$  also do not show any differences in the methodologies, as figure 3 shows. Thus, there is a general possible convergence of all six listed sampling methods, including the raw data, towards an optimal classification result. A reason for this result can be the stochastic initialization with the quantity  $I=100$ , by which a favourable starting position of the weights  $w$  for convergence is given.

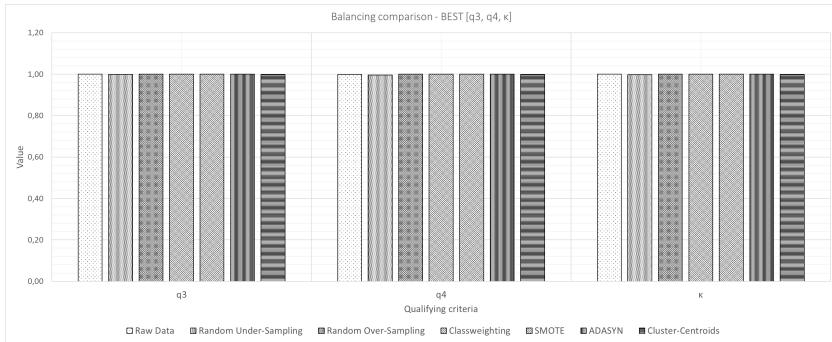


Figure 3: Balancing comparison of the best runs with [q3, q4, κ]

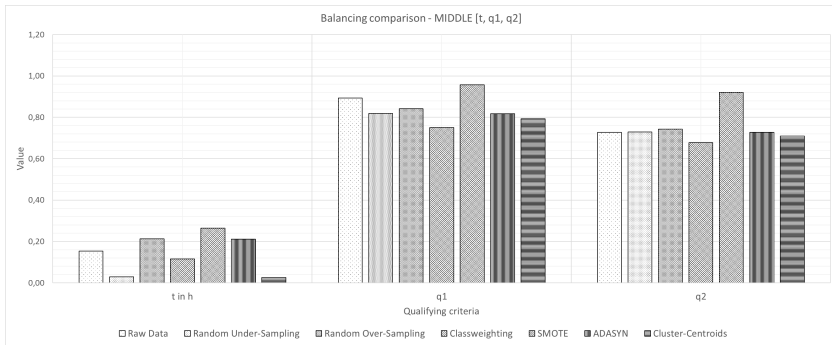


Figure 4: Balancing comparison of the middled runs with [t, q1, q2]

Representations of the best learning outcomes of a set of individuals can provide a static measure of convergence ability, but the statistical mean is a quantity-based measure of how often the set of individuals moves towards convergence. Therefore, the parameters  $t, q_1, q_2, q_3, q_4$  and  $\kappa$  are following formally averaged:

$$\bar{x} = \frac{1}{I} \sum_{i=1}^I x_i \quad (23)$$

$I$  represents the set of individuals and  $x$  is the formal representative of all used parameters.

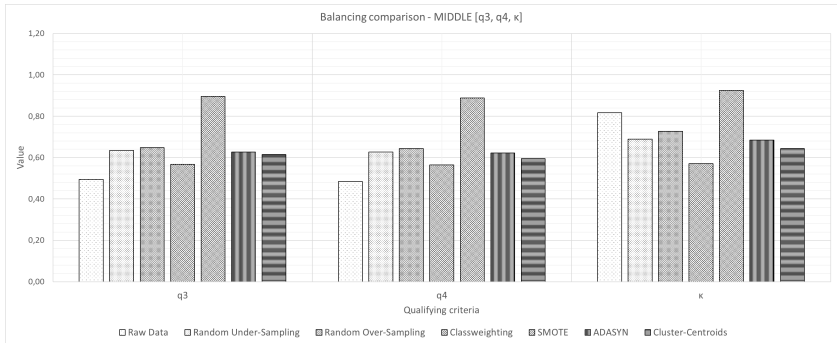


Figure 5: Balancing comparison of the middled runs with [q3, q4, κ]

By looking at the parameter  $t$  in figure 4 you can see that the behavior of the best runs is reflected. However, the set of balancers which need the longer calculation time is additionally supplemented by Random Over-Sampling. Noticeable with the parameters  $q_1$  and  $q_2$  is that the Classweighting performs worst and SMOTE as synthetic over-sampling is best. Interesting is the difference in raw data between the parameters  $q_1$  and  $q_2$ . Since  $q_1$  prefers the most represented class and  $q_2$  the less represented class, this means that the classifier can better learn the over-represented data with pure raw data.

The two parameters  $q_3$  and  $q_4$  are used to evaluate the worst recognized class and have a statement for the robustness of the classifier. Looking at these in figure 5 it becomes immediately clear that SMOTE produces very good results compared to the other methods and that the raw data provide a less robust classification. The results of the  $\kappa$  coefficient are similar to those of the parameter  $q_1$ . It shows that the class weights give a relatively poor result and again the SMOTE method generates the best results.

By considering the averaged coefficients, a quantity-related representation is achieved. The distributions of the parameter values are also to be analyzed. Thus, the help of the box plots is used. The boundaries of the box contained in the plots represent so-called quantiles, more precisely  $x_{0.25}$ ,  $x_{0.5}$  and  $x_{0.75}$ . Usually a so-called whisker is specified in connection with this, which is defined

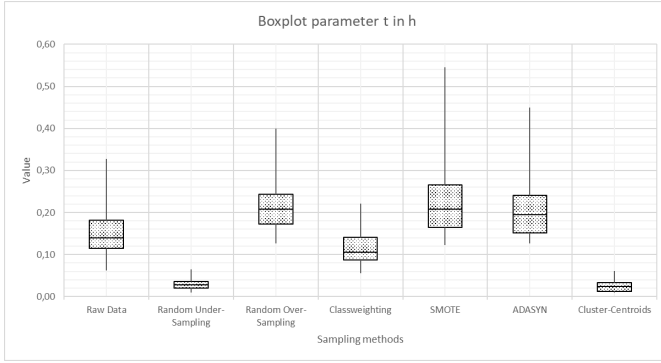


Figure 6: Boxplot of balancing methods regarding training time  $t$

in the following and non-uniform way:

$$w = 1.5 \cdot (x_{0.75} - x_{0.25}) \quad (24)$$

This whisker limits the values up and down. However, if the whisker falls below or exceeds the minimum and maximum values, these are assumed. This procedure was used in the following illustrations.

The variance of the  $t$  parameter shows, in figure 6, that the over-sampling procedure has an increased median including larger minima and maxima. This can be attributed to the quantity-related increase in samples. The under-sampling methods, on the other hand, scatter less, as well as the median is lower. A well-balanced means for this is the Classweighting.

The box plots of the parameter  $q_1$  in figure 7 show differences to the conventional appearance of these. First of all, it should be noted that this one, like the following box plots, is limited in  $[0; 1]$ . Furthermore, it can be seen that the scattering measure of all balancing procedures is similar. Only the raw data differ. It is interesting to note that the median  $x_{0.5}$  and the 3rd quantile  $x_{0.75}$  are very close together. This indicates a high frequency at this point. In addition, it can be seen that the median for the raw data is lower than for the balancing procedures.

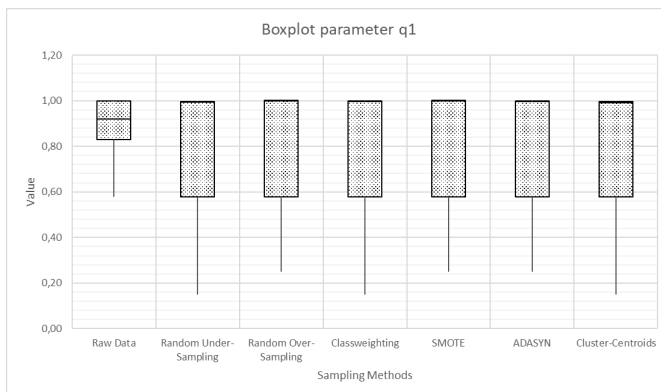


Figure 7: Boxplot of balancing methods regarding parameter  $q_1$

The parameter  $q_2$  shows similar behavior in figure 8 as  $q_1$ . However, since it prefers the underrepresented classes, there is more variation. Also, the classification is worse when teaching with raw data, which shows its median.

With  $q_3$  a robustness measure is represented in figure 9. The scatter is very high for all methods and covers the whole value range  $[0; 1]$ . For all sampling methods, however, the median is close to 1, which has a high frequency at this point. Only the raw data performs worse. Its median is close to 0.5. This indicates a balanced distribution.

Since  $q_4$  is also a measure for robustness, it can be assumed that a similar form of box plots as in figure 9 can be expected. This is confirmed with figure 10.

As a final parameter  $\kappa$  should be addressed. In previous plots this parameter showed equivalent behavior to  $q_1$  with respect to the best run and the averaged runs. It can be seen in figure 11 that all sampling procedures have a high degree of dispersion. For all of them  $x_{0.25}$  is close to 0.25. The minimum values also tend to be as low as 0. The median with the 3rd quantile is close to 1. This again indicates a higher frequency at this point. As an exception the individuals of the classifications with the raw data are shown. These scatter less, because their 1st quantile is around 0.7, but the median is also lower, 0.84 to be exact. This shows the clear influence of the balancing procedures on the stochastic initialization of the weights for the given dataset.

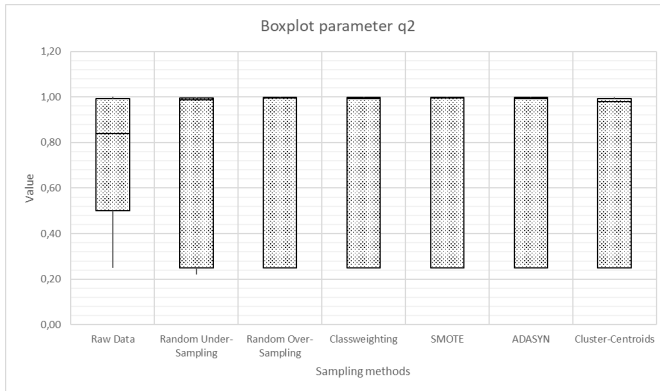


Figure 8: Boxplot of balancing methods regarding parameter  $q_2$

## 4 Conclusion

In this paper the aim was to investigate the influence of different balancing methods on audio data. In order to isolate the effect, it was decided to use synthetic data as a basis. With this data a predefined structure of a deep convolutional neural network was trained. The only variant part of the network was the random initialization of the weights.

With the representation in figure 2 and 3 statements concerning convergence can be made. All balancing methods show, from a static point of view, the convergence ability to solve the problem. The defined parameters all reach the value 1 in  $[0; 1]$ , except the calculation time. This time is proportional to the number of samples used in the learning process. In figure 4 and 5 the averaged values of the parameters is displayed. This shows a dynamic measure, since with each run, i.e. an individual, a different convergence is achieved and the average value is used to consider these individuals. At first it can be seen, that the calculation time is proportional to the given number of samples in the balanced datasets. Furthermore, pure raw data without balancers in the learning behavior prefer the overrepresented classes. With inverse logic this behavior is shown for the underrepresented data. The best middle balancing method is the SMOTE algorithm. This represents itself as extremely convergent with regard to all quality parameters  $q_1, q_2, q_3, q_4$  and  $\kappa$ , whereas the other procedures tend

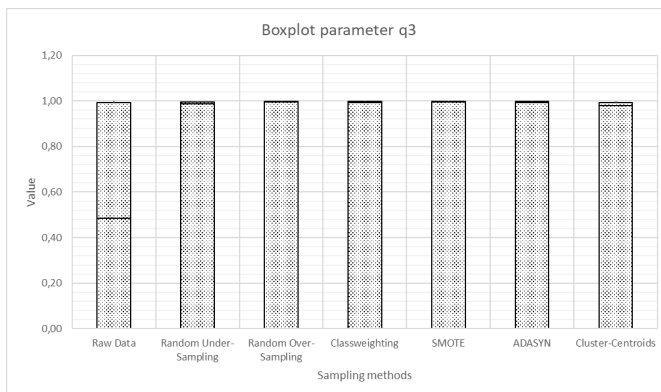


Figure 9: Boxplot of balancing methods regarding parameter q3

to be less optimal. With the analysis of the dispersion of the quality parameters, in figure 6 to 11, high variances are found in all balancing procedures. It shows that, in addition to the convergence, there is also a divergence for each algorithm due to pessimistic initialization of the weights. Likewise, the measure of dispersion itself is similar to most balancers per parameter and decides only on the raw data. A strong binarity between con- and divergence crystallizes.

From the results it can be concluded that the best balancing method is SMOTE. This of course only requires the application to the synthetic audio data. In the following work the effects on real measurement data have to be investigated. Since this contains noisy expert specifications in addition to the noise-superimposed signals, the results have to be awaited. Another effect could also occur when considering robustness as well as convergence.

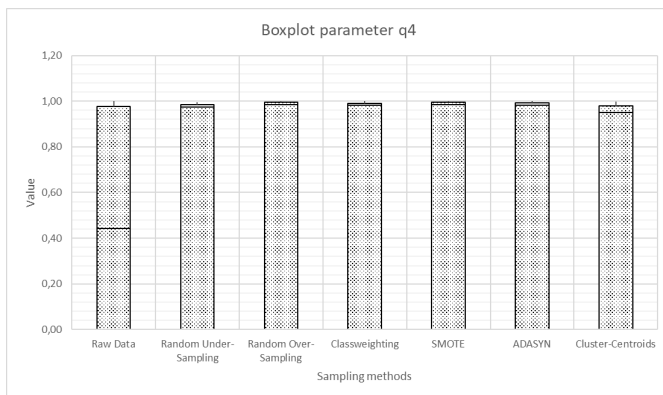


Figure 10: Boxplot of balancing methods regarding parameter q4

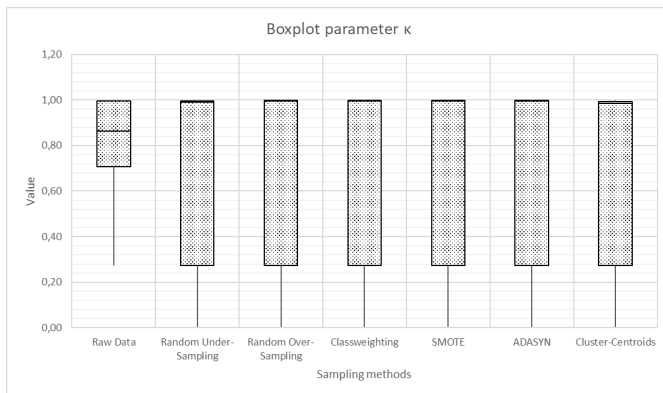


Figure 11: Boxplot of balancing methods regarding parameter k

## References

- [1] S. Hershey *et al.*: “CNN architectures for large-scale audio classification”. In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA* pp. 131-135 2017.
- [2] M. Trommer, M. Schneider, C. Walther: “Auswirkungen von ungleichverteilten und ungenauen Belehrungsdaten auf die Klassifikation der Support Vektor Maschine”. In: *Tag der Forschung 2014, Schmalkalden* pp. 35-52 2014.
- [3] Haibo He and Yang Bai and E. A. Garcia and Shutao Li: “ADASYN: Adaptive synthetic sampling approach for imbalanced learning”. In: *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)* pp. 1322-1328 2008.
- [4] V. S. Spelman and R. Porkodi: “A Review on Handling Imbalanced Data”. In: *2018 International Conference on Current Trends towards Converging Technologies (ICCTCT)* pp. 1-11 2018.
- [5] Maria Trommer: “Ein Beitrag zur Anwendung von Support-Vektor-Maschinen zur robusten nichtlinearen Klassifikation komplexer biologischer Daten”. In: *Beitrag zur Anwendung von Support-Vektor-Maschinen zur robusten nichtlinearen Klassifikation komplexer biologischer Daten* 2017.
- [6] T. Maciejewski and J. Stefanowski: “Local neighbourhood extension of SMOTE for mining imbalanced data”. In: *2011 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)* pp. 101-111 2011.
- [7] Pérez-Ortega, Joaquín AND Almanza-Ortega, Nelva Nely AND Romero, David: “Balancing effort and benefit of K-means clustering algorithms in Big Data realms”. In: *PLOS ONE* pp. 1-19 2018.
- [8] Haibing Wu and Xiaodong Gu: “Towards dropout training for convolutional neural networks”. In: *Neural Networks* pp. 1-10 2015.

- [9] X. Liang, X. Wang, Z. Lei, S. Liao, and S. Z. Li: “Soft-Margin Softmax for Deep Classification”. In: *Neural Information Processing* pp. 413-421 2017.
- [10] Andreas Wenzel: “Robuste Klassifikation von EEG-Daten durch Neuronale Netze”. In: *Robuste Klassifikation von EEG-Daten durch Neuronale Netze* 2005.