AnyNets: Adaptive Deep Neural Networks for Medical Data with Missing Values

Maria Kalweit[1], Gabriel Kalweit[1] and Joschka Boedecker[1,2]

[1]Neurorobotics Lab, Department of Computer Science, University of Freiburg, Germany
[2]BrainLinks-BrainTools, University of Freiburg, Germany

Abstract
During medical check-ups, commonly a large variety of measurements, procedures and medication prescriptions is collected. The type of measurements and procedures, however, may differ from patient to patient, hence a dataset can contain a large amount of incomplete samples with respect to the full set of considered features. Common work either use imputation methods (filling in the missing values with some default) or marginalization (ignoring samples with missing entries), which can lead to significant bias or limit generalization. In this work, we present a novel class of adaptive deep neural networks called AnyNets that can deal with missing values by design without the need for imputation. We show the advantage of AnyNets for both supervised and unsupervised learning on various medical datasets.

Keywords
Adaptive Deep Learning, Medical Data, Missing Values

1. Introduction

Learning from complex medical datasets is challenging due to their irregular, noisy and incomplete nature. Deep learning methods have shown great potential in a number of domains, however learning from incomplete data has been identified as very challenging [1]. Missing data can occur due to various reasons, e.g. because it got lost or it was never collected in medical check-ups, as shown in Figure 1. Further possible reasons are human errors or that they are difficult to collect, like measurements from biopsies [2]. In general, missing values can be categorized into three subtypes [3]: (1) missing completely at random (MCAR), if the absence is independent of the variables (2) missing at random (MAR), if the data is missing depending on the observed variables, and (3) missing not at random (MNAR), if missing values depend on observed and unobserved variables. Classical imputation methods rely on filling missing values with defaults (e.g. zeros) or values computed based on observable features, such as mean imputation (inserting the mean value of a feature into missing entries) and k-NN imputation (substituting a missing entry with its k-nearest neighbor) [4, 5]. Imputation of missing values, however, inherently changes the nature of a sample due to the assumptions being made which

Second Workshop on Artificial Intelligence for Function, Disability, and Health (IJCAI-21)
kalweitm@cs.uni-freiburg.de (M. Kalweit); kalweitg@cs.uni-freiburg.de (G. Kalweit); jboedeck@cs.uni-freiburg.de (J. Boedecker)

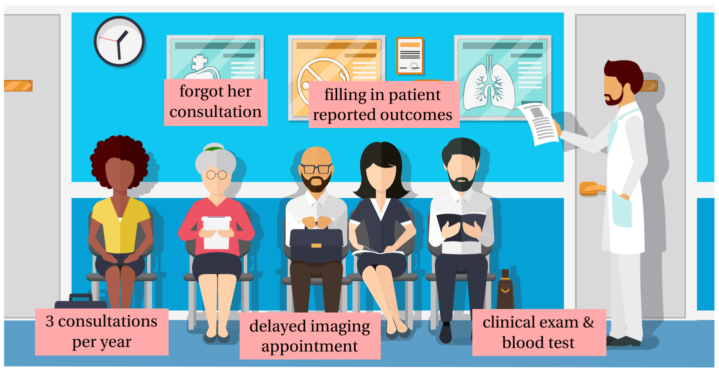
forgot her consultation
filling in patient reported outcomes
3 consultations per year
delayed imaging appointment
clinical exam & blood test

Figure 1: Irregular and noisy data in electronic medical records and registries are not trivial to handle with deep neural networks or classical machine learning methods due to a large number of different features and missing values.

may lead to wrong generalization or bias increase, especially in the MNAR case. It is therefore desirable to remove the necessity of imputation.

In this work, we focus on neural architectures which are able to process solely observed features as input in order to avoid the need of imputing missing values explicitly. Therefore, it is necessary to be able to deal with variable-sized lists of input features. Dealing with variable-sized lists of inputs was already done in prior work, e.g. with long short-term memorys (LSTMs) for time series [6] and in the medical field to deal with a variable number of visit and medication feature vectors [7]. In these feature vectors, however, missing values were still replaced by dummy values. In this paper, we aim at avoiding imputation completely, due to the reasons described above.

We therefore build upon the Deep Sets architecture [8], that projects objects to a latent space and pools the latent vectors via a permutation-invariant operator. In this work, we propose a novel class of adaptive deep neural networks, AnyNets, that learns a combined representation of all non-missing features and that is further able to account for missing values in this combined representation implicitly from the data. AnyNets project single non-missing features to a latent space using feature-specific encoders and pool the resulting list of latent vectors, using the sum as pooling operator to create a representation of the sample. This latent sample representation can then be used by a fully-connected network for classification, called AnyNets-Classifier. Further, we propose a novel adaptive deep neural autoencoder, adding feature-specific decoders for unsupervised learning tasks such as clustering, called AnyNets-Autoencoder. The scheme of AnyNets is shown in Figure 2.

2. Background

In this section, we fix notation and introduce the necessary theoretical background.

2.1. Missing Values and Imputation

We formally define the data as matrix $X = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n)$, where $\mathbf{x}_i \in \mathbb{R}^D$ with $D = D_1 + \cdots + D_p$ comprises a sample of p features, each of dimensionality D_j with $1 \leq j \leq p$. Missing values can be formalized as mask matrix $S = (\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_n) \in \{0,1\}^{n \times p}$ with

$$s_{ij} = \begin{cases} 1 & \text{if } x_{i,j} \text{ observed} \\ 0 & \text{if } x_{i,j} \text{ missing} \end{cases}.$$

We then define observed data as $\hat{X} = X \odot S + (1 - S) \odot \mathrm{na}$, where $\odot$ is the elementwise multiplication and na identifies missing values. Imputation methods replace missing values using projection functions ρ that impute default values. The projection functions for zero, mean and k-NN imputations, as well as the projection function proposed in this work, are defined as follows:

Zero imputation. This imputation method inserts zeros to represent missing values:

$$\rho(\mathbf{x}_i) = \mathbf{x}_i \odot \mathbf{s}_i + (1 - \mathbf{s}_i) \odot \mathbf{0}.$$

Mean imputation. This imputation method uses the mean $\tilde{x}_{\cdot,j}$ over non-missing entries of feature $1 \leq j \leq p$ over all samples of the observed data $\hat{X}$:

$$\rho(\mathbf{x}_i) = \mathbf{x}_i \odot \mathbf{s}_i + (1 - \mathbf{s}_i) \odot (\tilde{\mathbf{x}}_{\cdot,1}, ...\tilde{\mathbf{x}}_{\cdot,p}).$$

k-NN imputation. This imputation method uses the mean value from k-nearest neighbors in $\hat{X}$, where two samples are close if the features that are not missing are close (measured using the Euclidean distance in the subspace of non-missing features):

$$\rho(\mathbf{x}_i) = \mathbf{x}_i \odot \mathbf{s}_i + (1 - \mathbf{s}_i) \odot \mathbf{knn}(\mathbf{x}_i, \hat{X}).$$

No imputation. In this work, we avoid to use any imputation by considering only the set of non-missing features for every state $\mathbf{x}_i$:

$$\rho(\mathbf{x}_i) = \{\mathcal{F}_j(\mathbf{x}_i) | 1 \leq j \leq p \text{ and } \mathcal{F}_j(\mathbf{x}_i) \text{ observed}\},$$

where $\mathcal{F}_j(\mathbf{x}_i) \in \mathbb{R}^{D_j}$ denotes the entries of sample $\mathbf{x}_i$ corresponding to feature $\mathcal{F}_j(\cdot)$.

Based on the projected data set $\rho(\hat{X}) = (\rho(\mathbf{x}_1), ..., \rho(\mathbf{x}_n))$, a deep neural network can be used to either:

1. Classify the input data based on a given label set $Y = (y_1, ..., y_n)$, approximating a function $f(\mathbf{x}_i) \approx y_i$ with $\mathbf{x}_i \in \rho(\hat{X})$ and $y_i \in Y$.
2. Cluster the samples in k groups in an unsupervised fashion using a clustering algorithm.

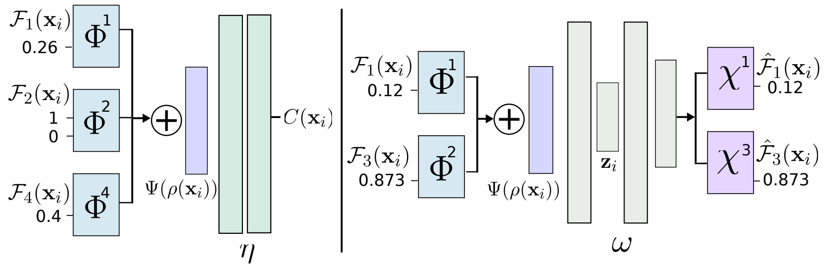
$\mathcal{F}_1(\mathbf{x}_i)$
0.26
Φ^1
$\mathcal{F}_2(\mathbf{x}_i)$
1
0
Φ^2
$\mathcal{F}_4(\mathbf{x}_i)$
0.4
Φ^4
$\Psi(\rho(\mathbf{x}_i))$
$C(\mathbf{x}_i)$
η
$\mathcal{F}_1(\mathbf{x}_i)$
0.12
Φ^1
$\mathcal{F}_3(\mathbf{x}_i)$
0.873
Φ^2
$\Psi(\rho(\mathbf{x}_i))$
$\mathbf{z}_i$
ω
χ^1
$\hat{\mathcal{F}}_1(\mathbf{x}_i)$
0.12
χ^3
$\hat{\mathcal{F}}_3(\mathbf{x}_i)$
0.873

Figure 2: Architecture schemes of AnyNets-Classifier for the sample input $\mathbf{x}_i = [0.26, (1, 0), \text{na}, 0.4]$, including numerical and categorical features, as well as a missing value (left) and AnyNets-Autoencoder for the sample input $\mathbf{x}_i = [0.12, \text{na}, 0.873, \text{na}]$, including two missing values and two numerical features (right).

For the imputation methods, it is straightforward to input the set of samples to a deep feed-forward neural network of input size $\sum_{i=1}^{p} D_i$, where D_i is the dimension of feature i ($D_i = 1$ for numerical features and $D_i = m$ for categorical features with m categories). Without imputation, a more sophisticated adaptive deep neural network architecture is required that can deal with a variable-sized set of input features for a batch of samples, which is explained in Section 3.

2.2. Deep Embedded Clustering

Deep Embedded Clustering (DEC) [9] performs clustering on the latent representation $\mathbf{z}$ of some autoencoder A by optimizing the loss $\mathscr{L} = \mathscr{L}_r + \gamma \mathscr{L}_c$, where:

$$\mathscr{L}_r = \sum_{x_{i,j} \in \rho(\mathbf{x}_i)} \|x_{i,j} - \hat{x}_{i,j}\|_2^2,$$

is the reconstruction loss and:

$$\mathscr{L}_c = \mathrm{KL}(P|Q) = \sum_i \sum_k p_{ik} \log \frac{p_{ik}}{q_{ik}},$$

the clustering loss. q_{ik} is the similarity between latent vector $\mathbf{z}_i$ and cluster center μ_k measured by the Student's t-distribution:

$$q_{ik} = \frac{(1 + \|\mathbf{z}_i - \mu_k\|^2)^{-1}}{\sum_k (1 + \|\mathbf{z}_i - \mu_k\|^2)^{-1}}.$$

p_{ik} is the target distribution:

$$p_{ik} = \frac{q_{ik}^2 / \sum_i q_{ik}}{\sum_k (q_{ik}^2 / \sum_i q_{ik})}.$$

In a first step, DEC pretrains the autoencoder by optimizing only the reconstruction loss ($\gamma = 0$) to compute latent representations in order to initialize the cluster centers μ_j via k-means [10]. Then, DEC jointly optimizes the clustering loss and reconstruction loss using $\gamma > 0$.

3. AnyNets

To deal with the above defined non-imputated dataset $\rho(\hat{X})$, we propose the deep neural network architecture AnyNets, which can deal with a variable-sized set of input features for a sample $\mathbf{x}_i$ by using p feature-specific encoder networks $\Phi^1, ..., \Phi^p$. The output vectors $\Phi^j(\cdot) \in R^F$ of the encoder networks Φ^j have the same length F. We additionally propose to share the parameters of the last layer over all encoder networks as done in [7] to enforce that all features are projected to the same encoded latent feature space. A combined representation of the non-missing features can be computed as:

$$\Psi(\rho(\mathbf{x}_i)) = \sum_{\mathscr{F}_j(\mathbf{x}_i) \in \rho(\mathbf{x}_i)} \Phi^j(\mathscr{F}_j(\mathbf{x}_i)),$$

using the sum as permutation-invariant pooling operator of all latent representations of features. Intuitively, we create higher-dimensional encodings of length F for all non-missing features and sum up all the encodings to gather information about all available features in a single vector of length F, which can be used as a latent representation describing the sample for classifier and autoencoder networks as described in the following. Thus, the networks can deal with a variable-sized list of input features, which makes them very suitable for data with many missing values.

3.1. AnyNets-Classifier

For classification tasks, an adaptive deep neural network architecture C can be defined, in which the latent representation of $\Psi(\rho(\mathbf{x}_i))$ is fed into a fully-connected network module η to compute the final prediction of the network with:

$$C(\mathbf{x}_i) = \eta(\Psi(\rho(\mathbf{x}_i))).$$

We optimize the cross-entropy loss:

$$\mathscr{L}(C) = -\sum_{i=1}^{n} \sum_{c=1}^{k} 1(c = y_i) \log(C(\mathbf{x}_i)[c]),$$

where $1(c = y_i)$ is the indicator function representing equality of c and y_i and $C(\mathbf{x}_i)[c]$ denotes the Softmax-probability of the network for the c^{th} class.

3.2. AnyNets-Autoencoder

For clustering, a deep autoencoder A can be defined by feeding the latent representation $\Psi(\rho(\mathbf{x}_i))$ into a fully-connected network module ω with hidden layer $\mathbf{z} \in \mathbb{R}^N$ in the middle layer. Introducing feature-specific decoder networks $\chi^1, ..., \chi^p$, the reconstructions of the input features $\mathscr{F}_j(\mathbf{x}_i)$ can be computed as:

$$\hat{\mathscr{F}_j}(\mathbf{x}_i) = \chi^j(\omega(\Psi(\rho(\mathbf{x}_i))),\ \forall \mathscr{F}_j(\mathbf{x}_i) \in \rho(\mathbf{x}_i).$$

The latent vector $\mathbf{z}_i$ for sample $\mathbf{x}_i$ can then be used to cluster the samples into k groups using clustering algorithms like DEC, explained in Section 2.2.

4. Datasets

We evaluate the classification performance of AnyNets on real-world medical datasets shown in Table 1 from the UCI repository[1] [11]:

- Kidney Disease: Classify patients suffering and not suffering from chronic kidney disease based on blood tests along with other measures.
- Diabetes: Classify patients with and without diabetes by diagnostic measurements.
- Mammographics: Predict the severity of a mammographic mass lesion (benign or malignant) from BI-RADS attributes along with the age of the patient.
- Breast Cancer: Cluster according to the severity of the cancer (benign or malignant). The dataset contains features characterizing the cell nuclei.
- Thyroid: Comprises demographic features, symptoms and laboratory tests, in order to cluster in normal, subclinical hypothyroidism and hypothyroidism.

The datasets contain from 5 to 24 features and up to 12% missing values. We evaluate the classification performance on Kidney Disease, Diabetes and Mammographics and evaluate the clustering performance on Breast Cancer and Thyroid, where we insert from 10-70% missing values artificially.

Dataset	#Samples	#Features	#Missing
Kidney Disease	400	24	10.54%
Diabetes	768	8	12.24%
Mammographics	961	5	3.37%
Breast Cancer	699	9	$10-70\%$
Thyroid	214	5	$10-70\%$

Table 1
Datasets for classification and clustering tasks.

5. Experimental Results

We evaluate the performance of AnyNets on three classification and two clustering tasks using the datasets described above. For all datasets, we apply 5-fold cross-validation to obtain mean accuracies and standard deviations over all folds. Hyperparameters are tuned using Random Search over 15 runs for all architectures, including the baselines, using the configuration spaces shown in Table 2. For preprocessing, all features are scaled in the range $[0,1]$. As activation function, we use rectified linear units (ReLU) for all hidden layers. For optimization we use Adam [12].

[1]https://archive.ics.uci.edu/ml/datasets.php

Model	Parameter	Config. Space
All	iterations	$[4000, 8000]$
	batch size	$[8, 16, 32, 64]$
	learning rate	$[1^{-4}, 1^{-5}]$
	hidden dim*,**	$[16, 32, 64, 128]$
FC-C	num layers	$[2, 3, 4]$
FC-AE	num layers*	$[2, 3]$
	latent dim.	$[2, 4, 8, 10]$
	η: hidden dim.**	$[16, 32, 64, 128, 512]$
AnyNet-C	Φ: hidden dim.	$[8, 16, 32, 64, 128]$
	Φ, η: num layers*	$[2, 3]$
AnyNet-AE	dim. of z	$[2, 4, 8, 10]$
	ω: hidden dim**	$[16, 32, 64, 128, 512]$

Table 2
Configuration spaces of the different approaches. (*) The hidden dimension for all fully-connected layers of all architectures. (**) If not denoted differently, for autoencoder networks, the hidden dimension and number of layers is used for both encoder and decoder modules. For DEC, $\gamma \in [0.1, 0.5, 1, 10]$ was used.

5.1. Classification Tasks

The classification results are shown in Table 3. The AnyNets-Classifier is outperforming the fully-connected neural networks in combination with mean, zero and k-NN imputation for all data sets. The best accuracy of 1.0 is achieved for the Kidney Disease dataset, where AnyNet is able to generalize perfectly to the test set. In Figure 3, we analyze the influence of missing feature values on the combined latent representation $\Psi(\rho(\mathbf{x}_i))$ for samples $\mathbf{x}_i$ in Kidney Disease with corresponding labels $y_i = 0$ (no chronic kidney disease) and $y_i = 1$ (chronic kidney disease). Single missing values for most of the features do not have a large impact on the latent representation, which means that they probably are missing at random and can be imputed implicitly by the neural network. Some features have a higher influence and change the meaning of the latent representation significantly.

Dataset	Mean	Zero	k-NN	AnyNets-Classifier
Kidney Disease	0.988 (±0.011)	0.985 (±0.015)	0.988 (±0.016)	**1.0** (±0.0)
Diabetes	0.767 (±0.030)	0.763 (±0.027)	0.763 (±0.027)	**0.772** (±0.038)
Mammographics	0.821 (±0.037)	0.822 (±0.040)	0.819 (±0.032)	**0.840** (±0.021)

Table 3
Mean accuracy and standard deviations for 5-fold cross validation over the medical datasets using supervised learning.

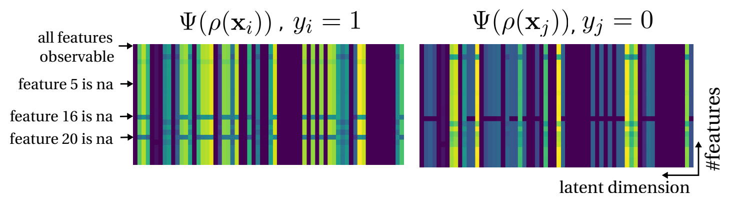

Figure 3: Visualization of the combined latent representations $\Psi(\rho(\cdot))$ for two samples $\mathbf{x}_i, \mathbf{x}_j$ with labels $y_i = 1$ (chronic kidney disease) and $y_j = 0$ (no chronic kidney disease) in Kidney Disease. The first row represents activations of the latent vector of size 64 for the sample x_i (left) and x_j (right) without any missing values (the brighter, the higher the activations), the k-th row the activations of the latent vector when setting the k-th feature as na. While a missing value for feature 5 does not have an impact (the latent vector looks similar to the vector in the first row), missing values for feature 16 and 20 change the feature representation significantly.

5.2. Clustering Tasks

To evaluate the clustering-performance of all approaches, we consider different percentages of inserted missing values for Breast Cancer and Thyroid. For clustering, we compute accuracies based on the predicted classes $\hat{y}_i$ (where all possible permutations of cluster IDs are considered) and the given ground truth classes y_i by:

$$\text{accuracy}(y_i, \hat{y}_i) = \max_{\text{perm} \in P} \frac{1}{n} \sum_{i=0}^{n-1} 1(\text{perm}(\hat{y}_i) = y_i),$$

where P is the set of all permutations and $1(\hat{y}_i = y_i) = 1$ if $\hat{y}_i = y_i$ and 0 else. The results for the Breast Cancer data set are shown in Table 4. AnyNets-Autoencoder outperforms all baselines, especially zero imputation by orders of magnitude for a high amount of missing values.

Missing Values [%]	Mean	Zero	k-NN	AnyNets-Autoencoder
10%	0.967 (±0.016)	0.963 (±0.012)	0.966 (±0.014)	**0.970** (±0.016)
30%	0.957 (±0.023)	0.913 (±0.008)	0.947 (±0.020)	**0.964** (±0.008)
50%	0.940 (±0.025)	0.707 (±0.119)	0.937 (±0.028)	**0.957** (±0.015)
70%	0.897 (±0.015)	0.621 (±0.089)	0.906 (±0.034)	**0.913** (±0.036)

Table 4
Mean accuracy and standard deviations for 5-fold cross validation over the Breast Cancer dataset with k=2 classes using DEC.

For Thyroid, shown in Table 5, the AnyNets-Autoencoder performs only slightly better than the k-NN imputation baseline, which might be due to the very small set of 214 samples.

Missing Values [%]	Mean	Zero	k-NN	AnyNets-Autoencoder
10%	0.743 (±0.116)	0.729 (±0.010)	0.753 (±0.103)	**0.771** (±0.106)
30%	0.711 (±0.123)	0.460 (±0.044)	**0.750** (±0.131)	**0.750** (±0.131)
50%	0.559 (±0.139)	0.452 (±0.066)	**0.706** (±0.067)	0.686 (±0.065)
70%	0.534 (±0.075)	0.432 (±0.046)	0.657 (±0.041)	**0.666** (±0.111)

Table 5
Mean accuracy and standard deviations for 5-fold cross validation over the Thyroid dataset with k=3 classes using DEC.

6. Discussion

The combined representation $\Psi(\rho(\mathbf{x}_i))$ of all observed features learned in AnyNets contains information of all the available features and implicitly is able to learn that certain features are missing. This stands in contrast to deep neural networks in combination with mean imputation or k-NN imputation and is of advantage if the absent values are related to the response data and not missing at random. For zero imputation, missing values can be represented as zeros, which is equivalent to setting the weights of the respective input neurons to zero; the bias, however, remains. The feature will thus still have an influence on the outcome, but possibly with an increased importance of the non-missing features. This effect can especially be seen in the results of unsupervised clustering. The remaining influence of the zero-imputed values changes the structure of the samples (also for MCAR and MAR variables) which leads to performance decrease in the unsupervised case for increasing amounts of missing values. The presence of labels in the supervised case alleviates this issue, hence the performance decrease is smaller.

7. Conclusion

In this paper, we introduced a novel class of adaptive neural networks called AnyNets which is capable of processing a set of variable input features. We showed the efficacy of AnyNets in the supervised and unsupervised case for various medical data sets and highlighted their advantages in their application to electronic medical records or registries which can contain many data types, ranging from continuous and categorical values to images, and commonly suffer from a significant amount of missing entries.

References

[1] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016. http://www.deeplearningbook.org.

[2] J. Yoon, W. R. Zame, A. Banerjee, M. Cadeiras, A. M. Alaa, M. van der Schaar, Personalized survival predictions via trees of predictors: An application to cardiac transplantation, PLOS ONE 13 (2018) 1–19. URL: https://doi.org/10.1371/journal.pone.0194985. doi:10.1371/journal.pone.0194985.

[3] R. J. A. Little, D. B. Rubin, Statistical Analysis with Missing Data, John Wiley Sons, Inc., USA, 1986.

[4] P. E. McKnight, K. M. McKnight, S. Sidani, A. J. Figueredo, Missing Data: A Gentle Introduction, Personnel Psychology, 2008.

[5] G. E. A. P. A. Batista, M. C. Monard, A study of k-nearest neighbour as an imputation method, in: HIS, 2002.

[6] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (1997) 1735–1780. URL: http://dx.doi.org/10.1162/neco.1997.9.8.1735. doi:10.1162/neco.1997.9.8.1735.

[7] M. Huegle, G. Kalweit, T. Huegle, J. Boedecker, Explainable ai in healthcare and medicine: A dynamic deep neural network for multimodal clinical data analysis, Studies in Computational Intelligence (2021). URL: http://dx.doi.org/10.1007/978-3-030-53352-6. doi:10.1007/978-3-030-53352-6.

[8] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, A. J. Smola, Deep sets, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems 30, Curran Associates, Inc., 2017, pp. 3391–3401. URL: http://papers.nips.cc/paper/6931-deep-sets.pdf.

[9] X. Guo, L. Gao, X. Liu, J. Yin, Improved deep embedded clustering with local structure preservation, in: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17, 2017, pp. 1753–1759. URL: https://doi.org/10.24963/ijcai.2017/243. doi:10.24963/ijcai.2017/243.

[10] J. Macqueen, Some methods for classification and analysis of multivariate observations, in: In 5-th Berkeley Symposium on Mathematical Statistics and Probability, 1967, pp. 281–297.

[11] A. Asunción, D. Newman, UCI Machine Learning Repository. http://www.ics.uci.edu/~mlearn/MLRepository.html, 2007. URL: http://www.ics.uci.edu/$\sim$mlearn/{MLR}epository.html.

[12] D. Kingma, J. Ba, Adam: A method for stochastic optimization, International Conference on Learning Representations (2014).