Stepwise Sample Generation *

Florian Bayeff-Filloff, Dominik Stecher and Kai Höfig

Rosenheim Technical University of Applied Sciences, Rosenheim, Germany
{florian.bayeff-filloff, dominik.stecher, kai.hoefig}@th-rosenheim.de
https://www.th-rosenheim.de

Abstract. Increasingly more industrial applications incorporate artificial intelligence based systems, which require a lot of data, e.g. captured from real machines or digital twin simulations, to train their models. But the data sets available or captured are often small, imbalanced or for specific use cases only. The most used approach to solve this problem efficiently are virtual sample generators. This type of program is commonly used to artificially construct new data from a small original data set, but tend to ignore weak correlation between features or even introduce false new correlations.
Therefore, we propose a novel stepwise synthetic sample generation approach which allows engineers and domain experts to include their domain knowledge directly in the sample generation process. They can judge how features depend on each other, which value range is plausible or acceptable and which features aren't relevant at all for getting better AI model approximation results and generating suitable synthetic data samples. In addition, we propose using simple machine learning models, such as support vector machines, to verify the synthetic data.

Keywords: Artificial Intelligence · Artificial Data · Software Development · Virtual Sample Generator

1 Introduction

The initial problem we attempt to solve is the large data requirement for currently applied, advanced Machine Learning (ML) methods such as deep learning, which often cannot be satisfied easily by Small and Medium Enterprises (SME) in cases such as rare defects, fault events, and low production volumes. Digital twins [3] can alleviate this issue, but tend to be complex to create and use, and come with increasing costs depending on the required fidelity of the simulation. However, useful information for the creation of a robust and effective AI model [10] are not only hidden inside captured samples - be that real-world measurements or digital twins data - but also in the minds of engineers and experts in the form of domain knowledge. To remedy this problem, we propose a virtual sample generator capable of including said knowledge and experience of

* Supported by Interreg Österreich-Bayern 2014-2020 as part of the project KI-Net (AB292)

technical experts familiar with the system at hand in AI training data generation by letting them directly influence the generation process of artificial samples.

The remainder of this paper is organized as follows: We discuss related work in section 2. In section 3 we describe our Stepwise Sample Generation approach in detail. Section 4 describes our evaluation procedure and shows first evaluation result. A closing summary and outlook for future works is given in section 5.

2 Related Work

It is well understood that larger data sets contain more useful information for the training of AI-based models and lead to an overall better model abstraction ability and quality [2, 5]. However, most real-world data sets at hand for SMEs are small in size, strongly imbalanced or meant for specific use cases only. In the last years researchers developed different approaches to gain better insights from small sample data. Some of the related studies are listed in the following.

2.1 Data Augmentation

In image classification, data augmentation [12] is a well-established approach to generate more data by altering existing images in different ways and thus gaining more information. Methods include scaling, flipping, colour and hue changes, filter operations and many more. Image 1 shows an example for classic image augmentations methods. However, such methods only work for image data, as rotating or flipping the measurements of a power meter results in nonsensical values.

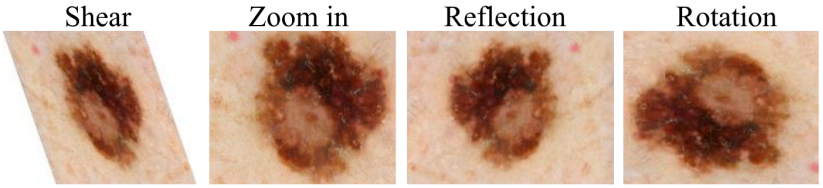

Fig. 1. Classic image augmentation provided by Mikolajczyk and Grochowski [7]

2.2 Virtual Sample Generator methods

For industrial applications, where data sets mostly consist of sensor data, the most common and broadly used approach are Virtual Sample Generators (VSG). These are algorithms designed to generate a large Synthetic Data (SD) set based on a small Original Data (OD) set. The base concept was first proposed by Poggio and Vetter [8, 9] as a method of sample augmentation, and has since then evolved

and been applied in many fields such as chemistry [16], medicine [14] or food monitoring [4]. In general VSG techniques fall into one of three categories: (1) Sampling-based, (2) Deep Learning-based and (3) Information Diffusion-based. In the following we list a few examples. Zhang [16] proposed a VSG based on the Isomap Algorithm, where they aim to find sparse populated data regions and interpolate the OD there to generate new data points. Zhu did something similar based on Locally Linear Embedding [18] and Kriging [17]. Yang [15] and Wedyan [14] proposed the usage of Gaussian distributed features to sample new feature values from. Another way of generating virtual samples are Generative Adversarial Networks. Douzas [1] proposed to extend this base concept by using conditions, and Li [6] added histograms to the training step to better match the SD to the OD.

The main problem we found with previous VSGs is that they often use a general purpose algorithm for all features within a given data set, regardless of their actual real-world correlations. This can lead to new correlations between features that shouldn't exist or ignore weak correlations, either because the overall data volume is too small or their representation to few for the algorithm to detect them reliably. In the absence of a data set large enough to properly represent all correlations between individual features, it is up to engineers and experts to provide this knowledge during the data generation process as well as to remove unwanted correlations.

3 Approach Overview

In this section we describe our SD generation approach. It was developed in Python as part of the Data Analysis & Augmentation Toolkit (DAAT), our software contribution to the Interreg KI-Net Project (AB292). This Toolkit currently consists of three parts: First, the Visual Analyser, for easy visual analysis of the given data set. Second, the Data Generator, which implements our data generation approach and several virtual sample generation algorithms. Third, the Verification Module, where simple ML models are trained and tested on OD and SD to verify the latter. We see this not only as a way to verify our approach, but as an integral part of the data generation process itself. In this paper we focus on our SD generation approach and the Verification Module.

The presented VSG can generate any number of synthetic samples based on a small OD set and improve the overall AI-model quality with limited data available. The flowchart of the general procedure employed by our approach is shown in figure 2. It consists of four main steps: (1) initializing the Generator, (2) selecting instructions for each feature defining dependencies and generation method, (3) executing the instructions in sequence, and lastly (4) verifying the generated data with the Verification Module.

3.1 Initialization

The first step is to initialize the Generator with the OD set and target label, if applicable. Currently only data sets consisting of numeric feature values are

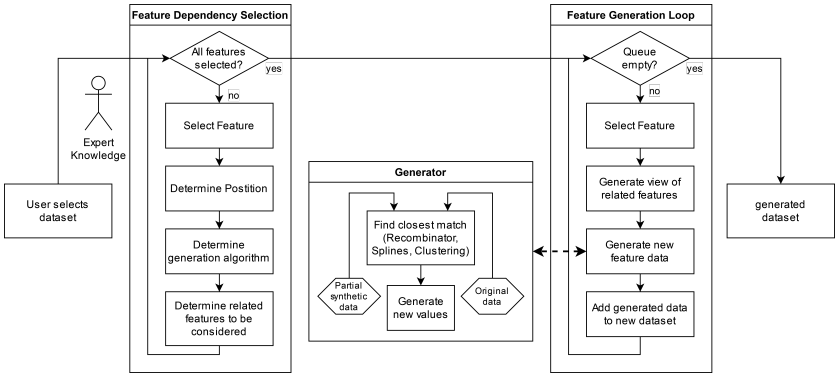

Fig. 2. Stepwise Sample Generation sequence

supported. The target feature can be defined as classification, regression, or none. This changes how the label is generated for the SD. For regression, the generated samples are matched with the n-nearest original samples, the mean is calculated and set as the new value. For classification, the data set is first separated by the target classes and the instructions are run on each class separately, setting the label as the current class. Otherwise, no label is generated. The support of other target labels and data set types, e.g., time series, is planed for future iterations.

3.2 Feature Dependency Selection

After initialization, instructions for each feature in the data set, except the target label, are defined and added to the generator pipeline. Each instruction tells the generator which of the available generation methods to use, which other features to use, as well as other method related settings, like the number of clusters to search for or lower and upper limits. Which features are dependent from each other and which generator method is the most fitting for each feature is considered domain knowledge. At this stage, the specific generation method can be chosen freely, but the instruction order is limited by the dependent features, as they must have been generated beforehand. This is because each generation method matches the partially generated SD to the OD set to generate values for the feature currently worked on.

Generation Methods As a proof of concept, our prototype implementation includes five simple VSG methods: Distribution, Spline, Recombine, Cluster, KNN Mean. Additionally, the marker None can be set to skip a feature. Each of the methods is implemented as a base class extension. This structure enables the implementation and use of self specified algorithms within the generator by the user. In the following we briefly describe each included generation method. Distribution takes a feature, determines the mean and standard deviation and then samples new values according to the found distribution. Spline/Griddata is meant for feature combinations which can be described by functions. Recombine

determines the k nearest original samples of the current partially generated synthetic sample and picks one value from them at random. Cluster runs a cluster algorithm on the OD set and determines the cluster the current synthetic sample lies within. It then samples a new random value from the cluster. KNN Mean is a simple interpolation algorithm, by taking the k nearest original samples and calculating the mean value.

3.3 Feature Generation Loop

Next, any number of synthetic samples can be generated and returned as a new SD set. For this the Generator runs trough the previously defined instructions in order. The values of the current feature are taken, a data set view of all specified dependent features is generated by filtering the OD set and the already partially generated SD set. These sub data sets are given to the selected generation method as parameters. Then the generation method is run and the resulting new values written in the partially generated SD set. After all instructions are run, the Feature Generation Loop (FGL) is finished, and the completed SD set can be retrieved from the Generator. Here we additionally give the option to retrieve the SD set combined with the OD seed.

Generator Setup Check Additionally, before starting the final FGL, the instructions can be evaluated. This sub step is optional but recommended and not shown in image 2. First, the instruction order is tested for inconsistencies and missing feature instruction definitions. Next, the instructions are executed on a small portion of the seed data set to generate an equally sized SD set. Finally, these data sets are given to the Verification Module, where they are evaluated against each other. The result is displayed as both table and graphs, providing an estimate on the instructions suitability and order validity in regards to the OD set. An example evaluation result is shown in figure 3, using the NASA KC1 (KC1) data set.

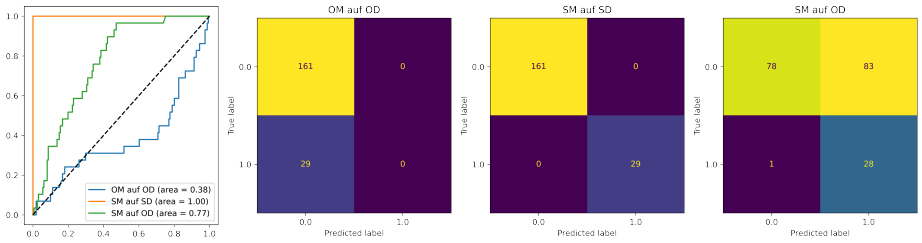

Fig. 3. KC1 data set - Setup Check using 30% OD

3.4 Verification Module

After the SD set is generated, it is advised to evaluate the new data set before proceeding with the time-consuming training of the final AI-model. For this, the DAAT includes a verification module using light-weight ML models such as Support Vector Machines (SVM) and Feed-Forward Artificial Neural Networks (FFANN).

For this purpose we split the OD and SD sets in training and test sets - typically 70:30. Next, we train two models with the exact same parameters on each training set and test each model on both test sets. This way, we can directly compare the performance of models trained on OD or SD sets. In a second step we train two new models each on the full data sets and use the other as test data. In both cases, we expect the SD-trained model to perform comparable or slightly better than the OD-trained model. We also expect the OD trained model to perform no worse on the synthetic test data.

4 Approach Evaluation

To demonstrate our proposed method, multiple experiments were performed on small data sets for classification problems. Using the KC1 data set [11] and Pima Indians Diabetes (PID) data set [13] we show our approach is capable of generating qualitative data samples - without trying to solve the actual problem such as fault detection - by comparing the performance and cross-evaluation results of ML methods on both OD and SD. The experiments were performed over 10, 100 or 1000 cycles, averaging the results to mitigate fluctuations introduced by random data sampling. While we acknowledge that better data sets exist, those typically lack the clear feature correlations or domain knowledge required.

Using PID, we generated two SD sets, one from 10% and one from 100% of the OD, labeled syn10 and syn100 respectively. Then three separate SVMs were trained on the resulting data sets and cross-evaluated over 1000 cycles. As shown in table 1 our approach can generate well data points using less OD as a seed.

Table 1. PID data set - Evaluation over 1000 Cycles using 10% and 100% data as seed

Model	orig			syn10			syn100		
Data set	orig	syn10	syn100	orig	syn10	syn100	orig	syn10	syn100
Accuracy	0.780	0.843	0.724	0.709	0.997	0.886	0.638	0.883	0.978
Precision	0.686	0.978	0.995	0.555	0.996	0.952	0.475	0.829	0.975
Recall	0.615	0.711	0.590	0.669	0.999	0.875	0.866	0.991	0.993
F1	0.649	0.808	0.739	0.594	0.998	0.904	0.613	0.900	0.984

Using KC1, 10, 20 and 40% from the training data were randomly sampled and labeled as gen. From these, SD sets were generated, such that they, combined

with their seed, were equally sized to the OD set and labeled as syn. Three simple FFANN were trained on the resulting data sets using class weights for balancing and evaluated using the original test data. As shown in table 2, our approach can generate data of equal quality as the OD from different sized seed data sets.

Table 2. KC1 data set - Evaluation for different OD ratios

Data set	orig.	gen			syn - 10CY			syn - 100CY		
Percentage	100%	10%	20%	40%	10%	20%	40%	10%	20%	40%
n OD	1475	148	295	590	148	295	590	148	295	590
Accuracy	0.686	0.601	0.621	0.664	0.773	0.753	0.734	0.752	0.764	0.762
Precision	0.269	0.239	0.224	0.254	0.359	0.274	0.289	0.339	0.347	0.331
Recall	0.638	0.748	0.598	0.645	0.537	0.489	0.602	0.564	0.496	0.500
F1	0.358	0.355	0.320	0.326	0.407	0.326	0.385	0.399	0.357	0.362

5 Conclusion and Future Work

In this paper we proposed a novel, modular and stepwise VSG approach. Our main goal was to provide an easy to understand example on how to incorporate domain knowledge within the data generation process. We tested our approach on the KC1 and PID data set and although the results are highly dependent on the instruction order, selected generation method and dependent features, our evaluations suggest that our approach can generate valid and qualitative SD.

For future iterations of our Toolkit, we plan to extend our approach. First, by support more data set types, like time series and non numeric sets, as well as the use of partially predefined data sets. Second, by extending the included five VSG methods with more sub types, e.g., Weibull and Poisson for Distribution, different cluster algorithms for Cluster, selection of different distance metrics etc., as well as more methods in general. We further plan to conduct case studies on actual data sets from industrial partners as well as university departments to assess the usefulness and scalability of our data generation approach on real world data.

References

1. Douzas, G., Bacao, F.: Effective data generation for imbalanced learning using conditional generative adversarial networks. Expert Systems with Applications **91**, 464–471 (jan 2018). https://doi.org/10.1016/j.eswa.2017.09.030
2. Hestness, J., Ardalani, N., Diamos, G.: Beyond human-level accuracy. In: Proceedings of the 24th Symposium on Principles and Practice of Parallel Programming. ACM (feb 2019). https://doi.org/10.1145/3293883.3295710
3. Kaur, M.J., Mishra, V.P., Maheshwari, P.: The convergence of digital twin, IoT, and machine learning: Transforming data into action. In: Internet of Things, pp. 3–17. Springer International Publishing (jul 2019). https://doi.org/10.1007/978-3-030-18732-3_1

4. Khot, L.R., Panigrahi, S., Doetkott, C., Chang, Y., Glower, J., Amamcharla, J., Logue, C., Sherwood, J.: Evaluation of technique to overcome small dataset problems during neural-network based contamination classification of packaged beef using integrated olfactory sensor system. LWT - Food Science and Technology **45**(2), 233–240 (mar 2012). https://doi.org/10.1016/j.lwt.2011.06.011
5. Kitchin, R., Lauriault, T.P.: Small data in the era of big data. GeoJournal **80**(4), 463–475 (oct 2014). https://doi.org/10.1007/s10708-014-9601-7
6. Li, W., Ding, W., Sadasivam, R., Cui, X., Chen, P.: His-GAN: A histogram-based GAN model to improve data generation quality. Neural Networks **119**, 31–45 (nov 2019). https://doi.org/10.1016/j.neunet.2019.07.001
7. Mikolajczyk, A., Grochowski, M.: Data augmentation for improving deep learning in image classification problem. In: 2018 International Interdisciplinary PhD Workshop (IIPhDW). IEEE (may 2018). https://doi.org/10.1109/iiphdw.2018.8388338
8. Niyogi, P., Girosi, F., Poggio, T.: Incorporating prior information in machine learning by creating virtual examples. Proceedings of the IEEE **86**(11), 2196–2209 (1998). https://doi.org/10.1109/5.726787
9. Poggio, T., Vetter, T.: Recognition and structure from one 2d model view: Observations on prototypes, object classes and symmetries. Tech. rep. (feb 1992). https://doi.org/10.21236/ada259735
10. Russell, S., Norvig, P.: Artificial Intelligence, Global Edition A Modern Approach. Pearson Deutschland (2021), https://elibrary.pearson.de/book/99.150005/9781292401171
11. Sayyad Shirabad, J., Menzies, T.: The PROMISE Repository of Software Engineering Databases. School of Information Technology and Engineering, University of Ottawa, Canada (2005), http://promise.site.uottawa.ca/SERepository
12. Shorten, C., Khoshgoftaar, T.M.: A survey on image data augmentation for deep learning. Journal of Big Data **6**(1) (jul 2019). https://doi.org/10.1186/s40537-019-0197-0
13. Sigillito, V.: Pima indians diabetes database. National Institute of Diabetes and Digestive and Kidney Diseases, The Johns Hopkins University, Maryland (1990), https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.names
14. Wedyan, M., Crippa, A., Al-Jumaily, A.: A novel virtual sample generation method to overcome the small sample size problem in computer aided medical diagnosing. Algorithms **12**(8), 160 (aug 2019). https://doi.org/10.3390/a12080160
15. Yang, J., Yu, X., Xie, Z.Q., Zhang, J.P.: A novel virtual sample generation method based on gaussian distribution. Knowledge-Based Systems **24**(6), 740–748 (aug 2011). https://doi.org/10.1016/j.knosys.2010.12.010
16. Zhang, X.H., Xu, Y., He, Y.L., Zhu, Q.X.: Novel manifold learning based virtual sample generation for optimizing soft sensor with small data. ISA Transactions **109**, 229–241 (mar 2021). https://doi.org/10.1016/j.isatra.2020.10.006
17. Zhu, Q.X., Chen, Z.S., Zhang, X.H., Rajabifard, A., Xu, Y., Chen, Y.Q.: Dealing with small sample size problems in process industry using virtual sample generation: a kriging-based approach. Soft Computing **24**(9), 6889–6902 (sep 2019). https://doi.org/10.1007/s00500-019-04326-3
18. Zhu, Q.X., Zhang, X.H., He, Y.L.: Novel virtual sample generation based on locally linear embedding for optimizing the small sample problem: Case of soft sensor applications. Industrial & Engineering Chemistry Research **59**(40), 17977–17986 (sep 2020). https://doi.org/10.1021/acs.iecr.0c01942