Analysis of parallel process in HVAC systems using deep autoencoders*

Antonio Morán†[1], Serafín Alonso[1], Miguel A. Prada[1], Juan J. Fuertes[1], Ignacio Díaz[2], and Manuel Domínguez[1]

[1]Grupo de investigación en Supervisión, Control y Automatización de Procesos Industriales (SUPPRESS), Esc. de Ing. Industrial e Informática, Universidad de León, Campus de Vegazana s/n, 24071, León, Spain, `http://suppress.unileon.es`
[2]Dept. de Ing. Elétrica, Electrónica, de Computadores y Sistemas, Universidad de Oviedo, Campus de Viesques s/n, Ed. Departamental 2, 33204, Gijón, Spain

Abstract

Heating, Ventilation, and Air Conditioning (HVAC) systems are generally built in a modular manner, comprising several identical subsystems in order to achieve their nominal capacity. These parallel subsystems and elements should have the same behavior and, therefore, differences between them can reveal failures and inefficiency in the system. The complexity in HVAC systems comes from the number of variables involved in these processes. For that reason, dimensionality reduction techniques can be a useful approach to reduce the complexity of the HVAC data and study their operation. However, for most of these techniques, it is not possible to project new data without retraining the projection and, as a result, it is not possible to easily compare several projections. In this paper, a method based on deep autoencoders is used to create a reference model with a HVAC system and new data is projected using this model to be able to compare them. The proposed approach is applied to real data from a chiller with 3 identical compressors at the Hospital of León.

*This is an author-created version of a contribution published in Communications in Computer and Information Science. The final publication is available at Springer via `http://dx.doi.org/10.1007/978-3-319-65172-9_2`.

This work was supported by the Spanish Ministerio de Ciencia e Innovación (MICINN) and the European FEDER funds under project CICYT DPI2015-69891-C2-1-R/2-R.

†a.moran@unileon.es

1 Introduction

Heating, ventilation and air conditioning (HVAC) systems represent about 50 % of the total consumption in the building sector, being the most energy-consuming equipment. It is equivalent to 10–20 % of the final energy consumption in developed countries [1]. Due to that exponential growth of HVAC energy use, policies and regulations focus on promoting energy efficiency of those building systems.

In order to understand how to improve the energy efficiency in buildings, it is necessary to monitor the HVAC systems. The working state of these systems should be analyzed to check the operation and detect malfunctions in those systems [2]. The use of advanced visualization tools can help to improve the efficiency of the systems [3]. However, the main problem creating these visualizations is that HVAC systems may comprise several modules and a vast number of variables each. In addition, these modules are composed of identical machines and elements working in parallel (according to the HVAC stages), so it is expected to have variables with the same evolution, hindering data visualization and comparison. For these reasons, it is necessary to reduce the number of variables, so that the processes can be visualized in an easy way and the visualization is consistent, allowing the data to be compared among the parallel processes.

Tools for visualizing multivariate systems have already been tested previously in order to draw conclusions about the behavior of the process [4]. Nevertheless, one problem of these techniques arises when projecting new or out-of-sample data points from the high dimensional space onto the low dimensional space. In this case, it is required to use specific algorithm modifications or run the algorithm again. Since these algorithms generally use random initialization, if we train again to include the new data the projection output changes, we cannot compare the results between reruns. Furthermore, it is impossible to deduce process values in the projection areas that do not display projected points, since these techniques are not bijective. Thus, it is necessary to use an additional interpolation technique with the projection method, making the creation of maps more complex and obtaining less accurate results [5]. As an example, the tools proposed in [4] combine data projection by means of the dimension reduction techniques such as Isomap, MDS, CCA, etc. [6] and an interpolation technique.

This paper proposes the use of a dimensionality reduction technique (Deep autoencoder) to project data while overcoming the aforementioned issues. Using this technique, it would be possible to project new data without retraining the algorithm, making easier the comparison among projections of different process. Furthermore, the projection algorithm is simpler, because it is a bijective method. Real data from a HVAC system at the Hospital of León, a chiller of 1407 kW comprising 3 identical subsystems (3 compressors), are used to test that dimensionality reduction technique.

This paper is structured as follows: The proposed approach is presented in Section 2. In Section 3, the testbed is described in detail. The experimental results are analyzed in Section 4. Finally, conclusions are drawn in Section 5.

2 Methodology

The main goal of our approach is to find a model that enables the projection of process data (composed by a large number of variables) onto a low dimensional space, where conclusions about the process behavior can be drawn in an intuitive manner. That model should allow the projection of new process data without modifying the distribution of the points already projected, in order to achieve a consistent comparison of the new points with the old ones, avoiding the need of model retraining. This kind of projection can be performed by means of *Deep Autoencoders* (DA) [7]. DAs do not only enable the projection of new points, but also define implicitly a *backward projection* from the low dimensional space to the high dimensional input space, allowing to infer input process data from areas of the output space where no previous projected data were available.

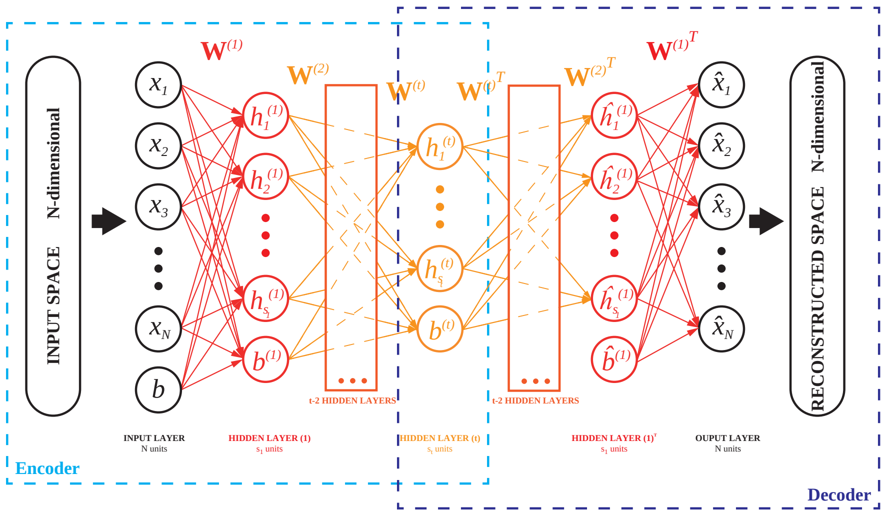
2.1 Deep autoencoders

An autoencoder is a type of neural network in which the output target of the network is set to be equal to the input; i.e., it is an unsupervised learning method which uses a back propagation algorithm for training. The autoencoder has at least three layers: an input layer, a hidden (encoding) layer, and a decoding layer. Since it is trained to reconstruct its inputs, the hidden layer is forced to learn a good representation of the inputs, so if the hidden layer is limited to a fewer number of neurons than that of the input layer, a dimensionality reduction will be performed [8]. Autoencoders have been widely used not only for dimensionality reduction, but also for feature extraction and denoising applications [9].

Despite single-layer NNs have been proved to achieve universal approximation, the number of units required for that purpose might be unfeasibly large and generalization is not guaranteed [10]. It must be noted that the *complexity* of the relationships involved in HVAC systems is one of the elements under consideration, because the efficiency of a HVAC system depends on many factors (internal and external variables).

Deep learning (DL) *models* have turned out to be good at discovering intricate structures in high dimensional data [11]. A deep autoencoder increases the number of hidden layers creating always a symmetric network in which the first half of the net represents the encoding, and the second half represents the decoding [12]. Figure 1 shows the structure of a deep-autoencoder. Choosing the structure of the autoencoder involves selecting the number of hidden layers (depth) and the number of units for each layer (width) and it is not a trivial task.

Recent works [13, 14] have showed that deep architectures allow computation of far more complex functions than shallow ones with a similar number of total units. The composition of layers allows identifying an exponentially growing number of input regions for large depths by means of successive space folding mechanisms, thereby enabling complex mappings by reusing pieces of computation. Moreover, the restriction imposed by the rigidity of the folding mechanism



INPUT SPACE N-dimensional
Encoder
Decoder
INPUT LAYER N units
HIDDEN LAYER (1) s_1 units
HIDDEN LAYER (t) s_t units
HIDDEN LAYER $(1)^T$ s_1 units
OUPUT LAYER N units
t-2 HIDDEN LAYERS
RECONSTRUCTED SPACE N-dimensional

Figure 1: Deep autoencoder structure

implicit in deep networks could be thought as a regularization mechanism that helps in achieving better generalization properties than shallow models [13].

Representation learning is another reason for choosing deep structures. The composition of successive nonlinear mappings that takes place in deep neural networks results in multiple levels of abstraction. The initial layers capture basic features of the input raw data, that are relevant for the problem, and subsequent mappings result in more abstract and also problem-relevant features built upon the former ones. In other words, DL networks turn out to learn feature detectors. Moreover, introspection into the internal layers of DL networks has revealed that the extracted features were surprisingly intuitive or meaningful in most cases.

The ability of DL networks in learning complex functions and their capability for representation learning, as argued above, have thereby suggested the use of deep autoencoders in this work for finding meaningful visual representations of parallel processes from the same HVAC system.

2.2 Visual analysis using autoencoders

We propose to project the data acquired from one of the process of a HVAC system, composed of several parallel units, in order to analyze its behavior. Using a set of variables, identical for all parallel units, allows us to obtain a *reference model*. Data from the reference unit are used to train a DA algorithm (see Fig. 2).

The proposed DA implements a “bottleneck” restriction of 2 output units in the *encoder* stage, because the aim is a mapping onto a 2D space. Such restriction forces the model to maximize the information flow in these two units

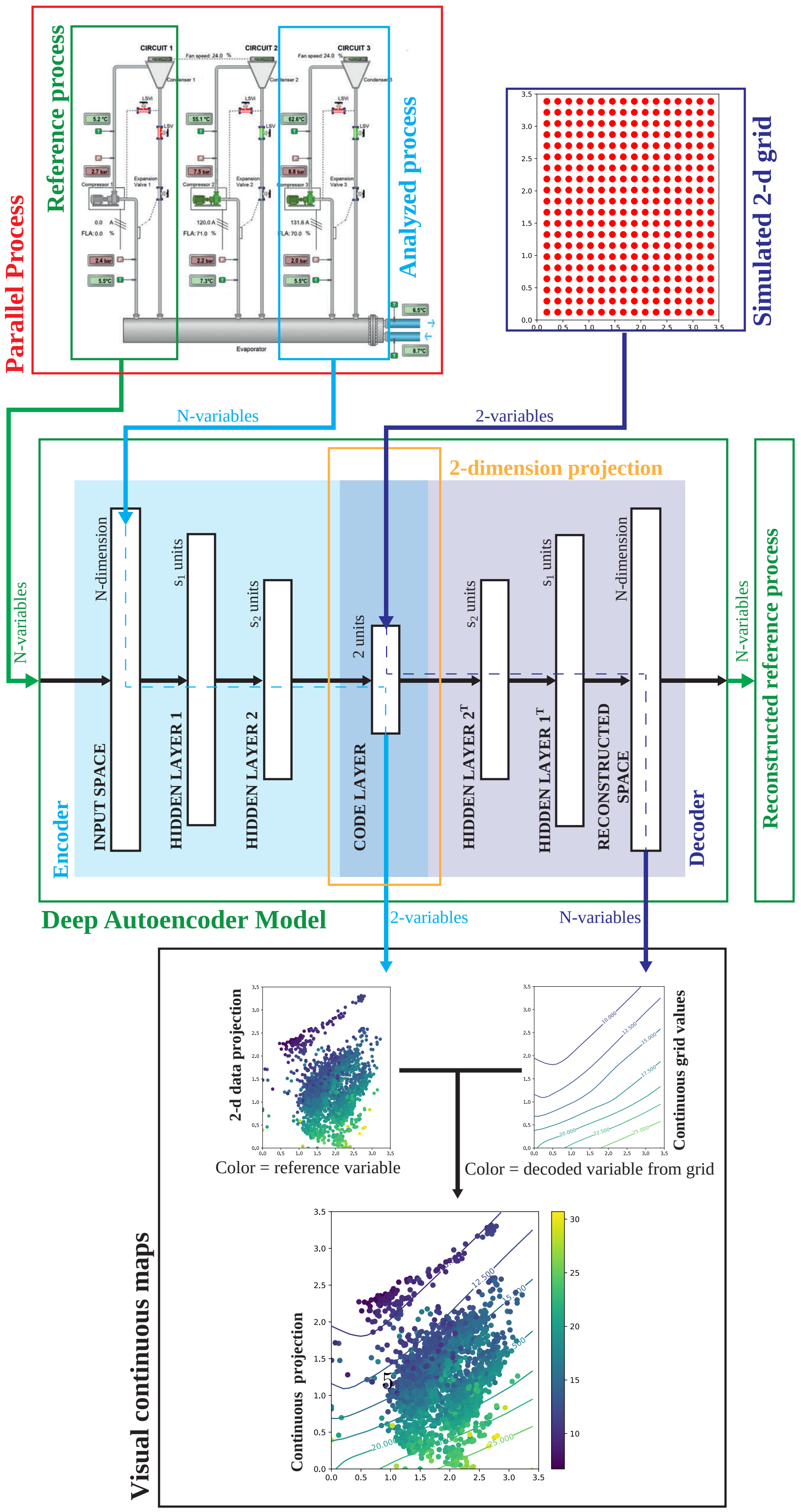


Figure 2: Methodology

in order to minimize the reconstruction error of the original process data carried out by the *decoder* stage.

- **Encoder**: once trained, this mapping of the high dimensional input space onto a 2D space allows to obtain projections for both *training* process data and new (*test*) process data.

- **Decoder**: this stage maps back 2D projections to high dimensional process data, thereby allowing a reconstruction of process data vectors from the projections. The implicit interpolation that appears in this stage can be used to build visual component planes of the process variables, by applying it to a regular grid in the 2D space that covers the extent of the projections.

This model can be used to achieve an effective visual comparison between different parallel processes. This can be done by obtaining the projection of data from the process unit used for training the model and then obtaining the projection of the other process unit using the same model. Both projections can be compared in the same visualization, considering that two projections lying in a similar region of the visualization space will correspond to similar values of the process variables. Thus, if all projections of two or more parallel process units span the same region in the projection, it reveals that their whole behavior is similar, while having projection clouds spanning different regions shows some kind of dissimilarity between the processes.

To provide context, it is proposed to complement the visualization with the information obtained from decoding the 2D regular grid points. Component planes corresponding to each process variable can be built by assigning to each point of the 2D grid a color corresponding to the value of its reconstructed a process variable, according to a color scale. Also, contour levels can be added to improve the visualization (see Fig. 2).

3 Experimentation system: Air-cooled chillers

The chiller plant at the Hospital of León is used as experimental system. Basically, that plant consists of a chilled water production subsystem and a distribution subsystem. Air-cooled and water-cooled chillers can be found in the production subsystem, together with valves, sensors and pumps needed to complement the chiller operation. There are 5 identical air-cooled chillers, comprising 3 internal refrigeration circuits each one, whose data are used as testbed for the approach.

Each air-cooled chiller (model Petra APSa 400-3) has a maximum cooling capacity of 400 tons (approximately 1407 kW) and includes 3 identical and independent refrigeration circuits (see Fig. 3). Each one is composed of a screw compressor, an electronic expansion valve (EEV), and 3 individual condensers in V form. A common evaporator is used for the 3 circuits. The compressor,

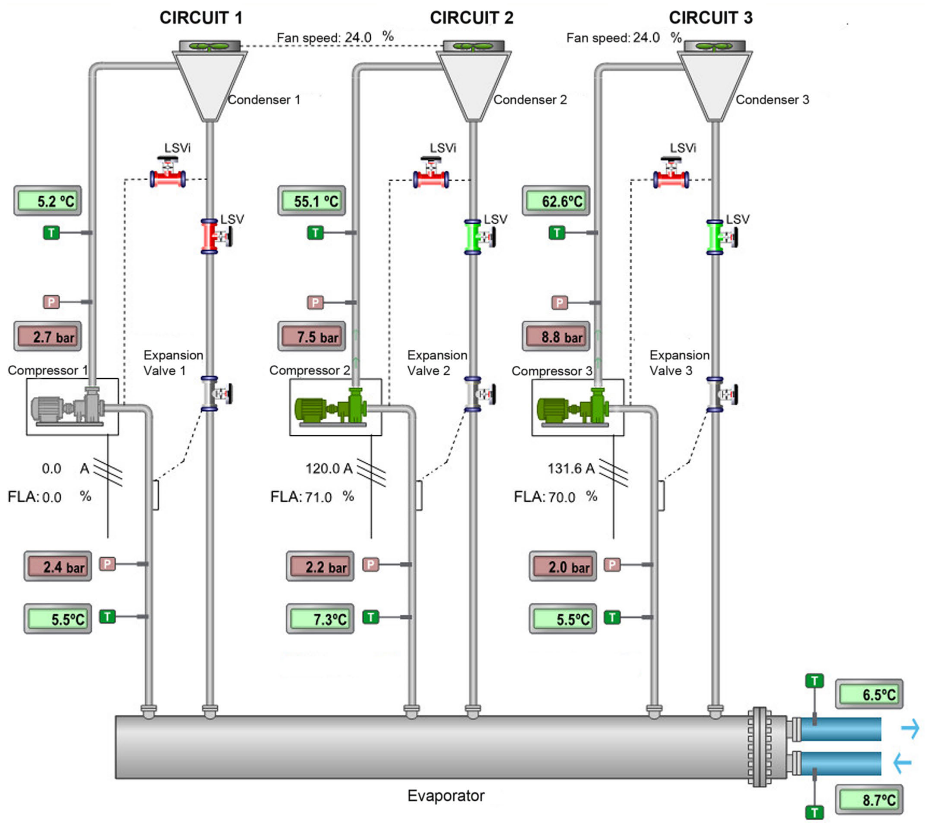


Figure 3: Air-cooled chiller refrigeration circuits.

driven by a three-phase induction motor (400 V; 109 kW), has a maximum displacement of 791 m^3/h of R134a refrigeration gas. Its capacity can be regulated between 50-100 % of maximum value by means of two auxiliary load and unload valves. The condensers have 16 fans of 1.5kW, driven by variable speed drives. Note that the compressor characterizes the electricity demand of the chiller (because it amounts to approximately 93 % of this demand). Each chiller requires external elements, such as a primary pump, which is driven by a variable speed drive to force water flow through the evaporator. Furthermore, an on/off valve is used to avoid water flow when the chiller is not running.

The control board acquires and controls several internal variables, being the most important ones listed in Table 1. It communicates with a central controller (Schneider Electric AS) which collects all chiller data using Modbus RTU protocol. Data is structured and stored in a SQLite database. Later, a Python service is used to preprocess raw data and build the training datasets.

Table 1: Internal variables to control each compressor circuit of the chiller.

Name	**Unit**
Evaporating temperature	$^{\circ}C$
Evaporating pressure	bar
Condensing temperature	$^{\circ}C$
Condensing pressure	bar
Compressor part load ratio	%
Compressor current	A
Chilled water leaving temperature	$^{\circ}C$
Chilled water entering temperature	$^{\circ}C$
Fan speed	%
Ambient temperature	$^{\circ}C$

4 Experimental results

To perform the experiments, a *deep autoencoder* has been trained in *Python* using *TensorFlow* [15], which is an open source library capable of building and training neural networks, and *Keras* [16], which is a high-level API running on top of either TensorFlow or Theano. Both libraries together let us program and train an autoencoder in an easy way. In addition, they can run on the *GPU* of a graphics card so that models and the projection of new points can be calculated faster, allowing the use of these techniques in almost real-time.

The data used in the experiments are acquired from a chiller at the Hospital of León, as described in the previous section. A model is obtained from the one of the compressors, which will be known from this moment on as Compressor1. This compressor has been selected as the reference unit, taking into account the information obtained in tests which were made beforehand on the chiller. These tests determined that it was the best calibrated compressor and the most optimal one. The variables used as input space are the ones described in the Table 1. This set of variables is acquired for each compressor. Once the reference model has been trained with the data from the Compressor1, the other two compressors, noted as Compressor2 and Compressor3 will be projected using this model.

The parameters used to train the autoencoder model have been selected manually due to the difficulty of using a method that allows identifying the most satisfactory projection. Although there are indexes such as the continuity and dissimilarity that can be used to measure the effectiveness of a projection [17], the complexity of the autoencoders implies a high number of tests that do not guarantee to obtain an optimal solution. For this reason, several projections are performed with different parameterizations and the one that provides a more intuitive visualization and comparison is selected according to our experience [6], since the final purpose of the projection is to obtain information through the visual analysis.

The resulting autoencoder consists of a 10-dimensional input layer and three intermediate layers of 128, 64 and 2 dimensions respectively. The last two-dimensional intermediate layer is the encoding layer. As for the decoding part

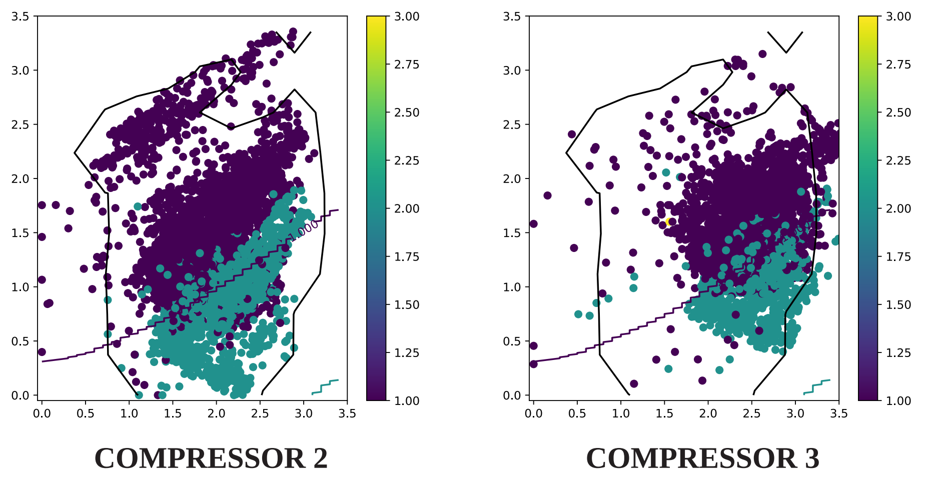


Figure 4: Data projection of the compressors over the space created with the model of the first compressor. The black line shows the projection area of the reference model (Compressor1).

of the autoencoder, it is symmetric, as explained before.

Figure 4 shows the result of creating a reference model with the data from Compressor1 and then, applying this model to the other compressors. The point colors are related to the number of compressors that work simultaneously in the chiller, which provide information about the machine working load. The black line marks the density of the model projection (Compressor1), i.e., this line gives the visual information of the contour within which most of the points of the model are projected. If the compressor to be compared works similarly to the model, its points are projected within the contour line. Otherwise, it indicates that the compressor does not have the same behavior as the reference one.

Figure 4 shows that Compressor2 is almost entirely projected within the contour, so it can be deduced that its operation is basically the same as that of Compressor1 (the reference one). However, in the case of Compressor3 projection, part of the dots are outside the contour and are projected to the right. Thus, it reveals a clear difference between the operation of Compressor3 and the reference (Compressor1). A thorough study of the chiller configuration proved that both the structure and parameters of the Compressor3 circuit have slight variations with regard to the other compressors. Specifically, the condenser of Compressor3 circuit is 17 % larger and has two fans more than the other circuits.

Figure 5 shows the behavior of several variables of the three compressors. It also shows the projection of the reference model (Compressor1) so that the three circuits can be analyzed and compared in more detail. In these maps, the color is representative of the value of the variable that is visualized (some of the visualized variables as ΔT are calculated from the temperatures used during

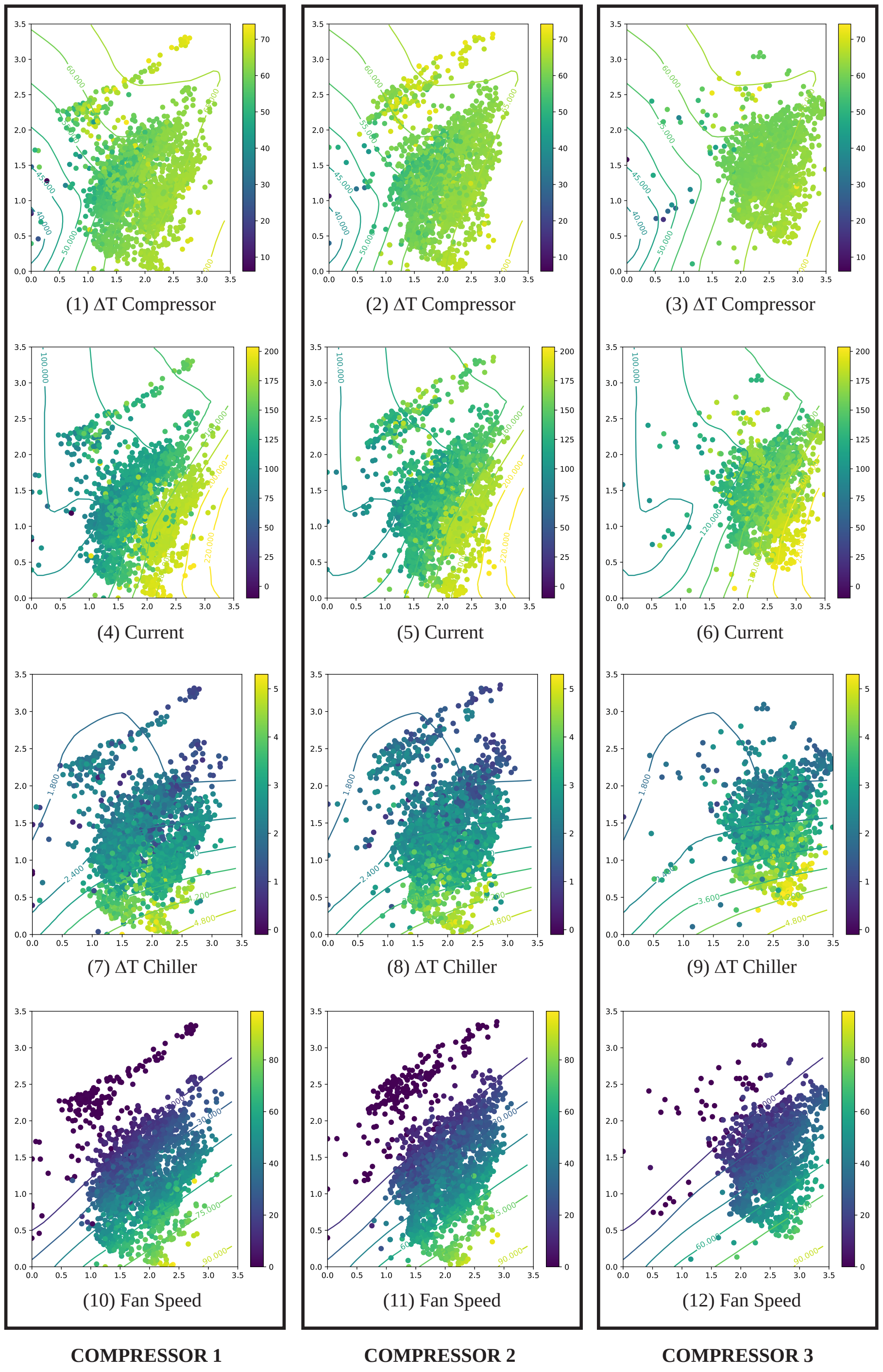


Figure 5: Different variables visualized for the three compressors of the chiller. The model is created with the first compressor and the lines shows the model reconstructed in the original space.

the projection) and the projection of the points is the result of applying the trained DA to each compressor circuit. The contour lines that appear on the projection come from applying the decoding model to the simulated grid that we used as a reference. The color of these curves has the same meaning as the color of the projected point, so they provide information about the distribution of the visualized variable values through the map.

Observing the projection of the points, we can check, using the maps, the similarity between the Compressor1 and 2, and the difference with the Compressor3. A priori, Compressor3 should take colder refrigeration gas (larger condenser), causing lower power demand. However, it can be observed that ΔT and the demanded current in the Compressor3 are a bit higher. The revision of parameters of the fan speed drive allowed us to explain this behavior. Speed limits on that configuration were discovered to try to compensate its associated larger condenser. As seen on Compressor3 projection maps, fan speed is a little lower than in the other two compressors. It can be also seen how the values of the remaining variables are quite similar.

5 Conclusions

This paper presents a new approach to reduce the data from multivariate identical parallel processes so that they can be visualized. The resulting two-dimensional visualizations can be compared in order to check whether the processes are similar, since differences in the working process result in a different projection.

A deep autoencoder is used to reduce the dimension of the data because it provides a series of advantages with respect to other existing techniques. It facilitates the projection of new points and it is also possible to carry out the inverse process obtaining the values of the space input when a grid in the output space is provided. The decoder component of the autoencoder has been used to interpolate in the input space using a simulated grid. This data is used to create a continuous projection on the maps to improve the information visualization. Using this feature, only a single model is needed, instead of having to incorporate additional techniques for interpolate the data. In addition, it is possible to project new points on the created model of the process without retraining the model.

In this paper, the technique was applied to a chiller with three parallel refrigeration circuits which are assumed to be identical. Since these circuits involve several variables, we use a dimension reduction technique to project compressor data and analyze their operation. The internal variables influence on the power demand of the compressor, which typically represents more than 90% of total energy demand in a chiller. Thus, compressor data are important to verify energy efficiency of a chiller.

We showed that it is possible to use deep autoencoders to perform a dimensionality reduction for industrial-oriented data visualization. We also showed that it is possible to make comparisons using the projected data between the model created with one process and the data of the others, checking easily whether the processes behavior are the same or not. It was found, using these projections, that the Compressor3 circuit of the chiller is different from the other two since the coordinates of the projected points are not the same.

As future work, this approach will be applied to the remaining air-cooled chillers at the Hospital of León (5 chillers with 3 compressors each). In addition, this methodology will be improved by using algorithms which make more practical the parameter selection and the training phase of the reference model of the process.

References

[1] L. Perez-Lombard, J. Ortiz and I. R. Maestre The map of energy flow in HVAC systems. *Applied Energy*, 88(12):5020-5031, 2011.

[2] L. Wang and S. Greenberg, J. Fiegel, A. Rubalcava, S. Earni, X. Pang, R. Yin, S. Woodworth, J. Hernandez-Maldonado Monitoring-based HVAC commissioning of an existing office building for energy efficiency. *Applied Energy*, 102:1382–1390, 2013.

[3] S. Meyers, E. Mills, A. Chen and L. Demsetz. Building data visualization for diagnostics. *ASHRAE Journal* 38(6), 1996.

[4] A. Morán, J. J. Fuertes, M. A. Prada, S. Alonso, P. Barrientos, I. Díaz and M. Domínguez. Analysis of electricity consumption profiles in public buildings with dimensionality reduction techniques. *Engineering Applications of Artificial Intelligence*, 26(8);1872–1880, 2003.

[5] L. Van Der Maaten, E. Postma and J. Van den Herik. Dimensionality reduction: a comparative. *J Mach Learn Res*, 10:66–71, 2009.

[6] J. A. Lee and M. Verleysen. Nonlinear Dimensionality Reduction. *Springer Publishing Company*, 2007.

[7] G. E. Hinton and R. R. Salakhutdinov. Reducing the Dimensionality of Data with Neural Networks. *Science*, 313(5786):504–507, 2006.

[8] Y. Wang, H. Yao and S. Zhao. Auto-encoder based dimensionality reduction. *Neurocomputing*, 184:232–242, 2016.

[9] L. Deng and D. Yu. Deep Learning: Methods and Applications. *Foundations and Trends in Signal Processing*, 7(3-4):197–387, 2014.

[10] I. Goodfellow, Y. Bengio, and A. Courville. Deep learning, 2016.

[11] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[12] P. Baldi. Autoencoders, Unsupervised Learning, and Deep Architectures. *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, 27:17–49, 2012.

[13] G. F. Montufar, R. Pascanu, K. Cho, and Y. Bengio. On the number of linear regions of deep neural networks. In *Advances in neural information processing systems*, pages 2924–2932, 2014.

[14] R. Pascanu, G. Montúfar, and Y. Bengio. On the number of inference regions of deep feed forward networks with piece-wise linear activations. *CoRR*, abs/1312.6098, 2013.

[15] Martín Abadi, Ashish Agarwal, Paul Barham et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, Software available from tensorflow.org `http://tensorflow.org/`, 2015.

[16] F. Chollet. Keras library. GitHub repository `https://github.com/fchollet/keras`, 2015.

[17] Venna, J., Kaski, S.: Comparison of visualization methods for an atlas of gene expression data sets. Information Visualization **6** (2007) 139–154