Iraki, Tarek; Link, Norbert

Article — Published Version

Generative models for capturing and exploiting the influence of process conditions on process curves

Journal of Intelligent Manufacturing

Provided in Cooperation with:
Springer Nature

Suggested Citation: Iraki, Tarek; Link, Norbert (2021) : Generative models for capturing and exploiting the influence of process conditions on process curves, Journal of Intelligent Manufacturing, ISSN 1572-8145, Springer US, New York, NY, Vol. 33, Iss. 2, pp. 473-492, https://doi.org/10.1007/s10845-021-01846-4

Journal of Intelligent Manufacturing
https://doi.org/10.1007/s10845-021-01846-4

Generative models for capturing and exploiting the influence of process conditions on process curves

Tarek Iraki[1] · **Norbert Link**[1]

Accepted: 14 September 2021

Abstract
Variations of dedicated process conditions (such as workpiece and tool properties) yield different process state evolutions, which are reflected by different time series of the observable quantities (process curves). A novel method is presented, which firstly allows to extract the statistical influence of these conditions on the process curves and its representation via generative models, and secondly represents their influence on the ensemble of curves by transformations of the representation space. A latent variable space is derived from sampled process data, which represents the curves with only few features. Generative models are formed based on conditional propability functions estimated in this space. Furthermore, the influence of conditions on the ensemble of process curves is represented by estimated transformations of the feature space, which map the process curve densities with different conditions on each other. The latent space is formed via Multi-Task-Learning of an auto-encoder and condition-detectors. The latter classifies the latent space representations of the process curves into the considered conditions. The Bayes framework and the Multi-task Learning models are used to obtain the process curve probabilty densities from the latent space densities. The methods are shown to reveal and represent the influence of combinations of workpiece and tool properties on resistance spot welding process curves.

Keywords Process modeling · Convolutional Neural Networks · Generative models · Multi-task learning · Low-dimensional process representations · Hyper-models · Machine learning

Introduction

A generic tool is proposed and devloped to automatically model manufacturing processes from time-series data of observable process quanities (process curves), especially to reveal how the process curves are shaped by process conditions such as tool properties or processed materials. The proposed method models the underlying conditional generative process producing the process curves. The tool can be used quite universally in intelligent process control or intelligent quality assessment, where process curves are the lead quantity such as in Model Predictive Control or where quality is measured as the deviation from an ideal curve. Such process control and quality assurance approaches can be automatically adapted to varying process conditions by just transforming the underlying process curves accordingly. This is enabled by the proposed machine learning methods, which reveal and generalize the dependencies of process curves on material and tool properties, and other conditions. This knowledge can also be used to understand the impact of varying conditions on the processes. The focus of this work is the development of such dedicated machine learning modelling methods from data and to show how they are applied to a real world process using the resistance spot welding process as a practical example.

A process is composed of a sequence of process states. Manufacturing processes change the state of a workpiece and terminate in a final state, in which the workpiece should have the desired properties. Dynamical equations exist for most physical and technical processes to model the time evolution of the process state $\mathbf{s}$, depending on the initial process $\mathbf{s}_0$ state (e.g. input material), on the process driving forces $\mathbf{u}$ and on the physical process boundary conditions $\mathbf{b}$. The quantities occurring in the basic dynamical equations are

✉ Tarek Iraki
tarek.iraki@h-ka.de

Norbert Link
norbert.link@h-ka.de

[1] Intelligent Systems Research Group (ISRG), Karlsruhe University of Applied Sciences, Moltkestr. 30, 76133 Karlsruhe, Germany

Published online: 07 October 2021

usually not completely available. Instead of the precise initial state only some properties $\mathbf{i}$ of the process input (e.g. Material type and geometry) are known. Also instead of the boundary conditions only machine characteristics $\mathbf{M}$ (such as tool type) are available in real processes and the process control adjusts only rough control parameters $\mathbf{p}$ instead of controlling directly the precise forces driving the process. The input properties and the machine characteristics represent the accessible fixed conditions governing a process, but are ambiguous with respect to the real process boundary conditions. This may lead to different process evolutions and outcomes under the same conditions. The outcomes are represented by some quality measures $\mathbf{q}$. The values of the process state variables are usually not directly measurable, but are only represented by a corresponding time series of measurements $\mathbf{m}(t_i)$, the process curve, represented by the vector of sampled values $\mathbf{z} = [\mathbf{m}(t_0), \ldots, \mathbf{m}(t_N)]^{\mathrm{T}}$. An emprical surrogate process model has therefore to be used in place of the precise dynamical model, relating the available condition quantities $\mathbf{c}$ with the state evolution represented by the process curve. The measured process curve correlates with the state evolution, but the exact mapping of the measurements to state variables is very frequently theoretically unknown or ambiguous. The process variables of a sufficiently restricted process sub-domain might populate only a suffienctly small sub-space for which a bi-jective mapping between state variables and process curves exists. In this case any representative sub-space of the process curves is also representative of the original process. The restriction of the process sub-domain is expressed by limited and small range of conditions.

Small sub-spaces are convenient for studying the impact of the conditions on the process curves and the relation of the curves with the process outcome. The impact can be captured statistically by modeling the condition-dependent conditional probability density of the process curves. The impact can be represented alternatively by a transformation between the densities with different conditions.

The paper proposes a method to find a well-suited process-curve sub-space representation (with so-called "features" or "latent variables"), which enables the approximate reconstruction of the original process curve, but also uses the most condition-sensitive features of the process curves. The resulting salient shape features minimize the reconstruction error of the process curves and also maximize the correlation with the conditions of the process. Every single process execution is represented by a point in the feature space. These feature points are then spread along dedicated directions according to the variation of corresponding condition values. Such a kind of a feature space is therefore optimally suited to model the influence of conditions and of the achieved quality on the process curves, thus relating all quantities of interest.

Our approach generates a feature space with sufficiently low dimensionality so that the probability density function can be estimated from feasible quantities of process curve samples. This density is used to infer the class conditional density via Bayes from the priors estimated by condition-classifiers using the features. This density is a generative model of the process under the considered condition and reveals its effect on the ensemble of process curves.

The effect of a change of conditions on the ensembles is found by estimating the latent space transformation, which maps the corresponding densities on each other. Generative models of all quantities of interest, which are derived from process curves (such as product quality or control strategy) can then be transferred to a different condition without setting up an explicit model for this condition.

The main benefit is a tremendous reduction of the number of experiments required to setup explicit models for the distributions of quantities of interest for each and every condition. It is only necessary to determine the dependency of such a quantity on process curves for one single condition. This model can be re-used for a new condition by transforming the underlying process curve distributions under the new condition to the original distribution and applying the model to it. This will deliver the distribution of the quantity under the new condition. From an experimental point of view, it is then sufficient to measure just the process curves only under the new condition (to determine the transformation between the new and the original process curve distributions), but not the quantity of interest itself. In resistance spot welding for instance, the quality is represented by the welding spot diameter, which is measured by destroying the joints in the lab, while the process curves are captured automatically on-line.

Our proposed method can also be used in future work to generate new generative models of process curves under previously unknown conditions by interpolation between transformation models if the dependency of the transformation model parameters on the condition values can be quantitfied. This generalization of the condition-dependency of the generative models is a kind of zero-shot learning (Larochelle et al. 2008; Socher et al. 2013) or hyper-modelling (Link et al. 2016; Reis et al. 2017).

Our contributions are the following:

- We create a low-dimensional feature space, where the representations of process curves under different conditions are spatially separated, through multi-task learning with Convolutional Neural Networks.
- We construct process-curve-generating models for different process conditions by estimating the corresponding probabilty densities in the low-dimensional latent space and transforming them to the original curve space by means of the multi-task-learned decoder.

– We estimate transformations of the latent features, which map the densities of different conditions on each other. The transformations allow the interpolation between the different conditions and thus condition-dependent morphing between process curve densities.
– We evaluate different neural architectures within our multi-task learning scheme with process curves from resistance spot welding. They are compared with a linear baseline method and among each other. The best architecture is then used to set up a generator model and to estimate a condition-dependent feature space transformation. Results of the the application of the generator and transformation are finally shown.

The paper is organized as follows: "Related work" section discusses the related work where the necessity of a new method to deliver a feature space with the required properties is revealed. The Bayesian construction of a process curve generator model and the representation of condition-dependent density transformations via a latent feature space representation are proposed in "Usage of condition-sensitive latent space process representations" section. In "Method for the determination of the latent feature space" section, a method for generating condition-sensitive, low-dimensional process representations is presented. The approach is instantiated for resistance spot welding processes in "Concept instantiations for resistance spot welding processes" section and finally evaluated with dedicated such processes in "Evaluation of different latent-space feature extractors for resistance spot welding process curves" section. A conclusion is drawn and an outlook given in "Latent-space representation and conditional generative and transformation models of resistance spot welding process curves" section.

Table 1 Summary of notation

Notation	Description
Variables	
$\mathbf{s}$	Process state
$\mathbf{s}_0$	Initial process state
$\mathbf{s}_{est}$	State, estimated from measurements
$\mathbf{u}$	External process driving forces
$\mathbf{b}$	Physical process boundary conditions
$\mathbf{i}$	Properties of the process input
$\mathbf{M}$	Machine characteristics
$\mathbf{p}$	Control parameters
$\mathbf{c}$	Process conditions
$\mathbf{c}', \mathbf{c}''$	Process condition instances
$\mathbf{q}$	Quality measures
$\mathbf{m}(t_i)$	Measured values of observables at time t_i
$\mathbf{z}$	Process curve $\mathbf{z} = [\mathbf{m}(t_0), \ldots, \mathbf{m}(t_N)]^{\mathrm{T}}$
$\mathbf{x}$	Latent features
Functions	
$h(\mathbf{z})$	Transformation of the representation space of the process curves
$f_{\mathrm{e}}(\mathbf{z})$	Encoder function
$f_{\mathrm{d}}(\mathbf{x})$	Decoder function
$f_{\mathrm{q}}(\mathbf{x})$	Estimator function
$f_{\mathrm{c}}(\mathbf{x})$	Classifier function

Related work

To our best knowledge there exist so far no approaches to extract condition sensitive generative models for process curves, as we are seeking for. There exist already machine learning approaches of time series generative models in other domains, but which are based on methods, which are quite data-intensive or where condition-dependency is hard to analyze, as discussed in "Time-series analysis of process data" section. Then we consider methods in the existing work to be used in a new approach, by which such drawbacks could be avoided. By reducing the representation space of process curves to a low dimensionality, the process curve space can be covered by smaller samples and the data efficiency be increased. As a positive side effect, visualization of the condition dependencies is possible as well. The *dimension reduction* methods are discussed in "Dimension reduction" section. *Variational and Conditional Variational Autoencoders* arrange the representation space of process curves such that represenations under the same conditions form clusters and can be interpreted w.r.t. the conditions. The corresponding related work is dicussed in "Variational and conditional variational autoencoders (VAEs and CVAEs)" section. Finally we summarize all this related work in relation to our specific task in "Summary of related work" section.

Time-series analysis of process data

Process data are by nature sequential and are captured as time series. Suitable representation methods and process models are not only important in manufacturing, but for processes in general such as weather, finance, speech and so on. Thus distance-based methods with a predefined distance measure such as Dynamic Time Warping (DTW) and Hidden-Markov-Models (HMM) operate directly with the data to either find similarities with given model templates (DTW) (Keogh and Ratanamahatana 2005) or to find a most probable state sequence as a cause of an observed time series (Ghahramani and Jordan 1997). The combination of DTW and the k-nearest-neighbor classifier allows to recognize time-series patterns (Rakthanmanon et al. 2012). In contrast, feature-based methods extract features that represent the pattern of the time series. Methods include, for example,

quantizing the features to form a bag-of-words (Lin et al. 2007), or extracting features of different scales to form a bag-of-features (Baydogan et al. 2013). However, these methods require very elaborate manual feature engineering based on expert knowledge.

The use of neural networks for time-series classification, especially by deep structures, allows to automatically find appropriate features, represented by the outputs of dedicated hidden layers. In Wang et al. (2017) the classification of time series using deep neural networks is investigated, based on the UCR Time Series Classification Archive (Chen et al. 2015) from different application domains, such as ECG, Natural Language Processing and so on. Convolutional Neural Networks with filter size decreasing from the input layer are applied in Dai et al. (2017) to time series, which represent acoustic signals by raw waveform data. These deep CNNs are used to generate features of the waveform input data for subsequent classification and serve to replace manual feature engineering, for end-to-end learning of the classification task. The input data are sequences with a vector of length 32,000 from which a lower number of features is extracted, which allow sufficient classification quality.

Recurrent Neural Networks (RNNs), namely Long-Short-Term-Memory (LSTM) networks have been established in modelling generators of (time) sequences of data (Graves 2013). An LSTM can generate process curves from initial values and a distribution of curves from adding noise to the initial and intermediate states. The models are directly represented by the weight values of the LSTMs, which form the representation space of the models. RNNs are enrolled into a finite number of time steps, where the weight values determine the dynamical behavior in the corresponding interval. If important temporal relations extend beyond this interval, the dynamics is not fully captured and generated sequences are not valid.

Only few machine learning approaches exist, which learn generative models of time series. In Yoon et al. (2019) the joint probability of time series and conditions is learned in the proposed framework. It consists of networks for embedding and recovery (auto-encoding) of the time series data, and a generator (operating on the latent space) and a discriminator for the sequences (GAN architecture, see Goodfellow et al. 2014). This requires the learning of the joint probabilities of the occurence of time series and conditions simultaneously. The sampling of the condition-time series product space requires a huge amount of data, which are rarely accessible with process data. Another time series geneartive model approach is presented in Esteban et al. (2017), which also uses a GAN architecture in combination with a Recurrent Network. These RNNs are conditioned on auxiliary information. The latent space in this approach only represents the evidence (unconditioned probability density), from which the LSTM recurrent network generates conditioned time series by using extra inputs representing the conditions. The goal was to generate training data for medical staff in ICUs. The condition-unspecific latent space does not allow to morph between generators under different conditions. The transfer of the learned knowledge to new conditions is therefore not possible.

Dimension reduction

Nonlinear dimension reduction methods are trying to represent data in a lower dimensional space, while preserving either the topology (TP) of the data (in mathematical sense), or the geometry (GP) reflected by the distance or angle between data. TP methods are Self-Organizing Maps (Kohonen et al. 2001), Locally-linear Embedding (Roweis and Saul 2000) und Laplacian Eigenmaps (Belkin and Niyogi 2001), while Multidimensional Scaling (Cox and Cox 2008), Kernel-PCA (Schölkopf et al. 1997) und Isomap (Tenenbaum et al. 2000) belong to the GP class. All the aforementioned methods assume that the data reside on a single manyfold in a subspace of the original data space. If the data populate multiple manifolds (eventually in different dimensions and possibly overlapping and intersecting), multi-manifold learning is required, such as K-Manifolds (Souvenir and Pless 2005) or DC-Isomap (Gao and Liang 2013), which decompose the manifold into intersecting-only and separated-only sub-manifolds respectively. All of these methods tend to find features, which optimally fulfil the method-specific criteria, but wich are not necessarily un-correlated or do not allow back-projection into the original space. Kernel versions of the Partial Least Squares (PLS) method allow the extraction of non-linear features and also maximize the correlation with some regression output variables (Rosipal and Trejo 2002). Kernel-PLS as a regression method would be able to create a latent space reflecting the influence of continuous quantities, but can not treat conditions with discrete values. Principal Function Approximators (PFAs) (Senn 2013) are especially constructed to realize a simultanouos invertable, non-linear dimensionality reduction of state variables and a regression of the extracted features with observables, which extends the linear methods of Partial Least Squares Regression (Vinzi et al. 2010) and Principal Component Regression (Merz and Pazzani 1999). Other than with (Kernel-)PLS, which builds upon variance, PFA features are formed by the bottleneck layer of an auto-encoder-like Artificial Neural Network and are not ordered according to relevance. The property of relevance-ordering is incorporated in Fischer et al. (2015) in neural auto-encoding via a hierarchy of single-neuron-bottleneck neural networks (HSB-NN), where each auto-encoder in the hierarchy represents the reconstruction residuum of its precedessor, and can also be trained to maximize the correlation with observables. As being regres-

sors like Kernel-PLS, these methods cannot treat processes with discrete-valued conditions.

Variational and conditional variational autoencoders (VAEs and CVAEs)

The dimension reduction methods discussed above do not yield latent space representations, which show separate distributions for different discrete conditions, because this is not enforced by the learning objective. Variational Autoencoders adress this problem by representing similar objects as being generated by a dedicated normal distribution (chracterized by a dedicated centroid and diagonal covariance matrix) in a latent space with chosen (low) dimensionality. Objects that form different similar groups are represented by multiplie normal distributions. The distributions are forced to be narrow and close to each other by including the Kullback-Leibler divergence in the cost function. The optimization eventually results in distinct normal distributions with different semantics (conditions in our case), which is found from the latent space representation of the condition-labelled sample data. Frequently, the distinct normal distributions are labelled accordingly, and linear interpolation in the latent space between the centroids of the distributions is used to generate objects (by the decoder), which are semantically interpolated (Hou et al. 2016). This is only possible, if the similarities between the objects in the original space reflects the semantics, because no other information than the structure of data itself is used for the training of VAEs. Neither the objective function used for training the VAEs nor the generic VAE structure enforce the capability to semantically interpolate in the latent space (Berthelot et al. 2019).

The problem of forcing VAEs to generate semantically conditioned object representations is addressed by the method of Conditional Variational Autoencoders (Sohn et al. 2015). The CVAEs enhance the object data by including the class memberships of the objects in their original description. This is achieved by representing the class-membership as a one-hot condition vector and concatenating this vector with the objects original data vector. In the resulting extended representation space the patterns of each class are represented in their own sub-space. Regular VAEs (structure and loss function) are then applied in this extended representation space to finally create the CVAEs. The resulting class-conditional densities in the latent space allow generating class-conditional object representations (by switching between the classes), but the densities are not spatially separated in the latent space. Therefore it is not possible to interpolate between classes in the latent space.

Summary of related work

The problems of data-driven representation of the condition-dependency of processes and finding process solutions for previously unseen conditions is generically tackled by Hyper-Models and zero-shot learning. A representation space is required for this purpose, where the process curve representations are arranged according to the conditions, and which has low dimensionality. To our knowledge, there is no data-driven approach so far to create such a representation for discrete- and continuous-valued conditions:

Dimension reduction of data is a well investigated scientific domain with a variety of methods for different goals, also embeddings, which allow the reconstruction of the data in the original space. Most of them yield a low-dimensional feature space, which represents the distribution of data well, but does not necessarily arrange the data points with respect to variations of the conditions, which influence the data. The latter property is incorporated by the Kernel-PLS methods and the PFAs and HSB-NN. PFAs and HSB-NN can be interpreted as following the idea of multi-task Neural Network training for the tasks of input pattern reconstruction (auto-encoding) and mapping on measurable observables. Replacing the observables by conditions would create a condition-sensitive feature space, but only for continuous-valued conditions.

VAEs are used to interpolate linearily between discrete-valued conditions, which is only possible, when the data in original representation space are already well arranged. CVAEs explicitely incorporate the conditions but in a way, which prevents semantic interpolation in the feature space.

Time-series analysis in the sense of extracting class information or predicting continuous quantities is a well established, but still very active scientific field. Recent approaches apply new methods, such as deep Convolutional Neural Networks (d-CNN) to enable end-to-end learning, thus avoiding feature engineering efforts. D-CNN have been very successful in image analysis, while applications to signals are still rare. No approach has been found, which uses d-CNN to extract features, which are also forced to allow reconstruction and bear information about conditions via multi-task learning at the same time. RNNs are frequently use in time-series analysis, but their capabilities seem to be too limited to capture long-range temporal relations in process curves.

The combination of GAN and RNN enables generative time-series models. Corresponding such approaches use a latent space representation for the distribution of time-series. We follow the idea of using a latent space, but employ a method, which forces a conditional structure of the latent space distributions and uses a decoder to generate conditional time-series.

Usage of condition-sensitive latent space process representations

Semantic process curve transformation and hyper-modelling

The generation of process curves $\mathbf{z}$ is subject to a stochastic process. The curve probability density $p(\mathbf{z}|\mathbf{p},\mathbf{c})$ represents the curve-generating stochastic process and is determined by the parameters $\mathbf{p}$ (controlling the forces on the process; usually adjusted by the process machine) and the conditions $\mathbf{c}$ simultaneously. In many practical cases the knowledge of the general influence of the conditions on the generator models of process curves is required, regardless of the chosen machine parameters $\mathbf{p}$. The condition influence is represented by the law of the transformation of the generator models (densities) according to a change in the condition values. The machine parameters are then considered as stochastic variables and the process curve densities are solely governed by the conditions: $p(\mathbf{z}|\mathbf{c})$. The process curve densities $p(\mathbf{z}|\mathbf{c}')$ with a certain condition $\mathbf{c}'$ are transformed into a density with new condition $\mathbf{c}''$ via $p(\mathbf{z}|\mathbf{c}'') = f(p(\mathbf{z}|\mathbf{c}'),\mathbf{c}'')$. We call this a semantic transformation of process curve densities. An application example is how the density of process curves of a reference process (determined in the laboratory) transforms into a new density, when the process is executed under different operating conditions than in the lab.

The density transformation can be expressed as a transformation h of the representation space of the process curves by $\mathbf{z}^* = h(\mathbf{z})$, such that $p^*(\mathbf{z}^*|\mathbf{p},\mathbf{c}') = p(\mathbf{z}|\mathbf{p},\mathbf{c}'')$. Such transformtion models $h(\mathbf{z})$ are estimated in "Latent-space representation and conditional generative and transformation models of resistance spot welding process curves" section in the latent space of the process curves. If instances of the transformation models between densities under different conditions are generalized into a general transformation $\hat{\mathbf{z}} = \hat{h}(\mathbf{z},\hat{\mathbf{c}})$ of a reference density with a condition-representing quantity $\hat{\mathbf{c}}$ as a parameter, a hyper-model of the process is generated.

Derivation of class-conditional density functions and generator models

Instead of training a generative model by means of a generator and a discriminator combined as opponents in a GAN, Bayesian inference can be used, if the proposed learning structure is implemented. The advantage of the Bayesian approach is the resulting explicit density function. This can be achieved by an encoder-generated data representation in a latent feature space and a multi-task processing scheme of the encoded data. The multi-task processing scheme must consist at least of a decoder and one or several classifiers or estimators, all of them attached to the encoder network. Such an architecture is depicted for process curves in Fig. 1.

The process curves are originally represented by vectors of sampled observable variable values $\mathbf{z} \in \mathbb{R}^N$ and transformed by the encoder according to a learned function $\mathbf{x} = f_e(\mathbf{z})$ to a latent feature space of vectors $\mathbf{x} \in \mathbb{R}^M$. The decoder transforms the feature vectors into approximate process curves $\tilde{\mathbf{z}} \in \mathbb{R}^N$. A soft-max trained classifier can be interpreted to deliver the estimated posterior probability $\tilde{P}(c_i|\mathbf{x})$ of having class c_i, when feature vector $\mathbf{x}$ is observed. If the dimensionality M of the latent feature space is small enough that the sample gives a good coverage of the underlying distribution, the probability density of the feature vectors $\mathbf{x}$ can be estimated as $\tilde{p}(\mathbf{x})$. This can be achieved by transforming all K sampled process curves (under all conditions) $\{\mathbf{z}_k\}_{k=1}^K$ to the latent feature space representations $\{\mathbf{x}_k\}_{k=1}^K$ and estimating an appropriate density function (i.e. multi-modal Gaussian). The priors of the classes can be estimated as $\tilde{P}(c_i)$ by the relative frequencies of the class instances. Then the class-conditional densities $\tilde{p}(\mathbf{x}|c_i)$ can be estimated via Bayes inference:

$$\tilde{p}_{\mathbf{x}}(\mathbf{x}|_i) = \frac{\tilde{P}(c_i|\mathbf{x})\,\tilde{p}(\mathbf{x})}{\tilde{P}(c_i)} \tag{1}$$

The density $\tilde{p}(\mathbf{x}|c_i)$ can be used to set up a generative model of instance of $\mathbf{x}$, when the cumulative distribution function calculated from $\tilde{P}(\mathbf{x}|c_i)$ is used to map an i.u.d. random variable r to get a sample vector $\mathbf{x}_s = \tilde{P}_{-1}(r)$ (Larson and Odoni 1981).

But a generator of process curves $\mathbf{z}$ instead of their feature representations $\mathbf{x}$ is desired. By construction we can find a generator of the approximate process curves $\tilde{\mathbf{z}} \approx \mathbf{z}$ using the decoder, which is represented by the learned function $\tilde{\mathbf{z}} = f_{\mathrm{d}}(\mathbf{x})$, which transforms feature representations to the original process curve space.

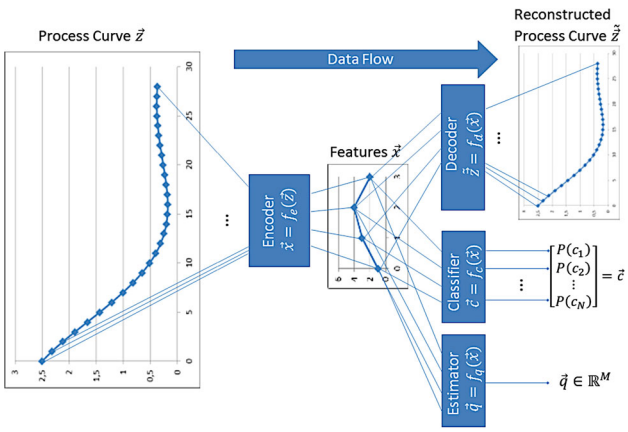

Fig. 1 Multi-task processing scheme with a latent feature space representation for deriving generative models and for semantic process curve transformations modeling

$$\tilde{p}_{\tilde{\mathbf{z}}}(\tilde{\mathbf{z}}|c_i) = \tilde{p}_{\mathbf{x}}(f_d(\mathbf{x})|c_i)\left|\frac{d\mathbf{x}}{df_d(\mathbf{x})}\right| \tag{2}$$

Using the auto-encoder property, we can approximate

$$\frac{d\mathbf{x}}{df_d(\mathbf{x})} = \frac{d\mathbf{x}}{d\tilde{\mathbf{z}}} \approx \frac{d\mathbf{x}}{d\mathbf{z}} = \frac{df_e(\mathbf{z})}{d\mathbf{z}} \tag{3}$$

And insert (3) into (2) to finally get

$$\tilde{p}_{\tilde{\mathbf{z}}}(\tilde{\mathbf{z}}|c_i) = \tilde{p}_{\mathbf{x}}(f_d(\mathbf{x})|c_i)\left|\frac{df_e(\mathbf{z})}{d\mathbf{z}}\right| \tag{4}$$

The estimated and approximated density $\tilde{p}_{\tilde{\mathbf{z}}}(\tilde{\mathbf{z}}|c_i)$ is a generator model of process curves. This can be used to study the general influence of a process condition on the process curves. The influence of a continuous-valued condition or outcome can be revealed from data in a similar way only if the condition is discretized in intervals and a classifier for the condition-value intervals is used to find the present condition range.

The continuous condition estimator as shown in Fig. 1 can be used to derive the distribution of the continuous-valued condition $\mathbf{c}$ or outcome $\mathbf{q}$, which is related with the distribution of process curves of a discrete condition (Eq. 1), which is derived by Bayesian inference as described above. For the example of resistance spot welding, this might be the distribution of the quality measure "welding spot diameter" (continuous condition) of different welding guns and metal sheet combinations (discrete conditions).

With the estimator for a single quantity q represented by $q = f_{\mathrm{q}}(\mathbf{x})$, the distribution of q can be found by

$$\tilde{p}_q(q|c_i) = \tilde{p}_{\mathbf{x}}(f_q(\mathbf{x})|c_i)\left|\frac{d\mathbf{x}}{df_q(\mathbf{x})}\right| \tag{5}$$

With the method presented so far the statistical properties of the process under different conditions can be modeled and the influence of such conditions be studied in terms of moments of the modelled distributions.

The resulting density functions under different conditions can be used as well to estimate transformation functions of the latent space (and thereby of the curve representation space), which transform the densities from one into the other, as discussed in the previous section.

Method for the determination of the latent feature space

Multi-task learning objective function

The proposed approach shown in Fig. 1 automatically derives features, which allow the optimal reconstruction of process curves and also capture maximum information about process conditions. The features can therefor represent the process curve deformation caused by different process conditions. Also the effect of different process outcomes on the shape of the process curves is represented by the features.

The encoder transformation of the data into the feature space is learned via multi-task learning together with the involved models of the decoder, classifier and estimator tasks, which process the data in the feature space. Each of the Neural Network mapping functions of Fig. 1 is parametrized by its weight values, which forms the parameter vector $\boldsymbol{\lambda}$: With the encoder output $\mathbf{x} = f_{\mathrm{e}}(\mathbf{z}, \boldsymbol{\lambda}_e)$ we get the latent space representation of $\mathbf{z}$. Using the latent space representation as input we get the outputs of the total multi-task network:

$$\begin{aligned}\tilde{\mathbf{z}} &= f_{\mathrm{d}}(\mathbf{x}, \boldsymbol{\lambda}_d) = f_{\mathrm{d}}(f_{\mathrm{e}}(\mathbf{z}, \boldsymbol{\lambda}_e), \boldsymbol{\lambda}_d),\\ \mathbf{c} &= f_{\mathrm{c}}(\mathbf{x}, \boldsymbol{\lambda}_c) = f_{\mathrm{c}}(f_{\mathrm{e}}(\mathbf{z}, \boldsymbol{\lambda}_e), \boldsymbol{\lambda}_c),\\ \mathbf{q} &= f_{\mathrm{q}}(\mathbf{x}, \boldsymbol{\lambda}_q) = f_{\mathrm{q}}(f_{\mathrm{e}}(\mathbf{z}, \boldsymbol{\lambda}_e), \boldsymbol{\lambda}_q)\end{aligned} \tag{6}$$

The parameters of all networks are simultaneously optimized, which means, that for the combined parameter vector $\boldsymbol{\lambda}_{\mathrm{all}} = [\boldsymbol{\lambda}_{\mathrm{e}}, \boldsymbol{\lambda}_{\mathrm{d}}, \boldsymbol{\lambda}_{\mathrm{c}}, \boldsymbol{\lambda}_{\mathrm{q}}]^T$ the optimum $\boldsymbol{\lambda}^*_{\mathrm{all}}$ is seeked for all process K curve samples $\mathbf{z}_i$ to deliver the corresponding $\mathbf{c}_i$ and $\mathbf{q}_i$ values and to reproduce $\mathbf{z}_i$ as good as possible, which is expressed by the search for the minimum of a loss function for the training sample set $\{\mathbf{z}_i, \mathbf{c}_i, \mathbf{q}_i\}_{i=1}^K$:

$$\boldsymbol{\lambda}^*_{\mathrm{all}} = argmin_{\boldsymbol{\lambda}_{\mathrm{all}}} J(\boldsymbol{\lambda}_{\mathrm{all}});\ J(\boldsymbol{\lambda}_{\mathrm{all}}) = \sum_{i=1}^{K} J_i(\boldsymbol{\lambda}_{\mathrm{all}}) \tag{7}$$

The loss function is composed of the weighted loss terms of all tasks, where γ_d is the weight of the decoder loss, γ_c the weight of the condition classifier loss and γ_q the weight of the outcome estimator loss.

$$J_i = \gamma_d J_{i_d} + \gamma_c J_{i_c} + \gamma_q J_{i_q} \tag{8}$$

The reconstruction loss of the decoder is the L^2 norm of the difference:

$$J_{i_d} = (\mathbf{z}_i - f_{\mathrm{d}}(f_{\mathrm{e}}(\mathbf{z}_i, \boldsymbol{\lambda}_{\mathrm{e}}), \boldsymbol{\lambda}_{\mathrm{d}}))^2 \tag{9}$$

The classifier loss is chosen as soft-max cross entropy Λ:

$$J_{i_c} = \Lambda(\mathbf{c}_i, f_{\mathrm{c}}(f_{\mathrm{e}}(\mathbf{z}_i, \boldsymbol{\lambda}_{\mathrm{e}}), \boldsymbol{\lambda}_{\mathrm{c}})) \tag{10}$$

The esitmator loss is again the L^2 norm of the difference:

$$J_{i_q} = (\mathbf{q}_i - f_{\mathrm{q}}(f_{\mathrm{e}}(\mathbf{z}_i, \boldsymbol{\lambda}_{\mathrm{e}}), \boldsymbol{\lambda}_{\mathrm{q}}))^2 \tag{11}$$

An optimizer is seeking the optimal set of all parameters $\boldsymbol{\lambda}^*_{\text{all}}$:

$$[\boldsymbol{\lambda}^*_{\text{e}}, \boldsymbol{\lambda}^*_{\text{d}}, \boldsymbol{\lambda}^*_{\text{c}}, \boldsymbol{\lambda}^*_{\text{q}}] = argmin_{\boldsymbol{\lambda}_{\text{all}}} \sum_{i=1}^{K} \gamma_{\text{d}}(\mathbf{z}_i - f_{\text{d}}(f_{\text{e}}(\mathbf{z}_i, \boldsymbol{\lambda}_{\text{e}}), \boldsymbol{\lambda}_{\text{d}}))^2 + \gamma_{\text{c}}\Lambda(\mathbf{c}_i, f_{\text{c}}(f_{\text{e}}(\mathbf{z}_i, \boldsymbol{\lambda}_{\text{e}}), \boldsymbol{\lambda}_{\text{c}})) + \gamma_{\text{q}}(\mathbf{q}_i - f_{\text{q}}(f_{\text{e}}(\mathbf{z}_i, \boldsymbol{\lambda}_{\text{e}}), \boldsymbol{\lambda}_{\text{q}}))^2 \quad (12)$$

The optimization also comprises and yields the optimal parameters of the process curve feature space transformation $\boldsymbol{\lambda}^*_{\text{e}}$, giving $\mathbf{x} = f_{\text{e}}(\mathbf{z}, \boldsymbol{\lambda}^*_{\text{e}})$.

General solution architecture

The optimization of the multi-task objective function (12) yields a low dimensional representation space with features, which are sensitive to the process conditions. Chosing a low-dimensional feature space forces the generalization of the features and lends itself to visualization. The proposed multi-task learning scheme of Fig. 1 allows to reveal and describe the influence of conditions/outcomes on the process curves by means of feature space analysis. These components are:

- The encoder, which transforms the process curves to their low-dimensional feature space representations and therefore enables the probability density estimation,
- The condition classifiers to estimate the condition probability posteriors depending on the latent-space features,
- The estimators to incorporate the relation between latent-space features and process outcomes, and
- The decoder to allow the reconstruction of class-conditioned curve probability densities.

The transformation models of all these components are determined from adapting the corresponding model parameters with a sample data set of sampled process curves. The curves are sampled by measuring the observables during process executions under different conditions. A representative sample requires that the condition space is sampled with sufficient density. After each process execution, the respective process outcomes are measured.

The multi-task objective function is optimized with these sample data. The optimization corresponds to the application of a machine learning methodology called "Multi-Task-Learning" (MTL) (Caruana 1997). It has a common feature extraction and multiple, subsequent task processing networks, each for a different task (see Fig. 2). MTL is a method to train a single, composed neural network for the simultaneous solution of different tasks. In the layers above the bottleneck they have common structures which are reused.

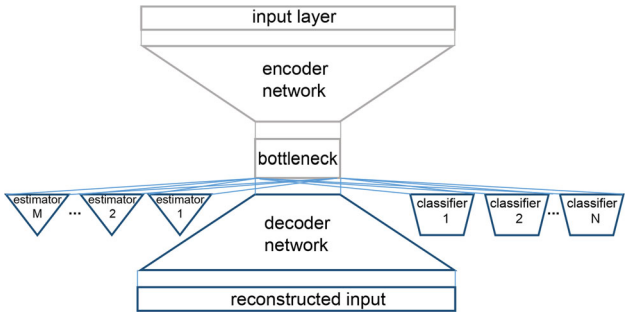

Fig. 2 Mulitask learning scheme

The layers below the bottleneck represent separate specializations for the different tasks. MTL is pursuing the approach of training all tasks in parallel with each other at the same time to allow a solution structure for one task to use the information of the solution structures for the other tasks. The tasks are of one of the following categories: (1) classification in the case of conditions and outcomes of discrete values, (2) estimation in the case of conditions and outcomes of continuous values and (3) the reconstruction (decoding) of the process curves to enable the feature extraction via auto-encoding.

Non-linear auto-encoding is usually achieved via bottleneck Neural Networks (DCNNs in our case), which are forced to learn to reproduce the input patterns at the output as precisely as possible with only a few numbers of bottleneck neurons (see Fig. 2). The activations of the bottleneck neurons are the feature values representing the input, resulting from the encoding by the preceding layers.

These activations are then used as input to the successive decoder layers, where the last layer delivers a pattern similar to the input. The training of the multi-task structure is performed by optimization according to Eq. 12 which is realized by Algorithm 1 below, where the function execute-task() also comprises the optimization of the encoder and task-specific parameters.

Algorithm 1 Multi Task Learning

```
for all epoch in epochs do
    if epoch mod number-of-tasks equals task-number then
        execute-task()
    end if
end for
```

Time-convolutional Neural Networks are proposed to be used for the encoder network because of the advantages of d-CNN for feature extraction found by Dai et al. (2017) and because of the fact that process curves can be thought of a composition of time-localized curve-shape primitives. Another motivation to propose the use of d-CNN is their success in image analysis, where they have been able to

significantly improve the results in segmentation, object detection and classification.

The most common CNN architectures align several convolutional layers, including the Relu activation function f(x) = max(0, x), which are followed by pooling layers. This pattern is repeated until the input data is reduced to a small representation. In addition, a subsequent processing by fully connected layers is common. The last fully connected layer generates the output data. Thus, the generic pattern for such CNN architectures is defined by:

Input → [[Conv → Relu]*N → Pool?]*M → [Fc → Relu]*K → Fc

Concept instantiations for resistance spot welding processes

The proposed concept of the previous chapters is instantiated to reveal the influence of conditions and outcomes on the process curves of resistance spot welding processes. Different Neural Network topologies are investigated and compared with each other and with a PCA feature extraction as non-neural baseline.

For the feature extraction, both, multilayer perceptrons and convolutional neural networks, are used. The goal is to fulfil all tasks satisfactorily with only a few features. Two, three, four and eight features will be use in the evaluation experiments.

In the context of hyper-parameter optimization of the Neural Networks, the method grid search is used. Based on the analysis of the hyper-parameters, the following decisions are made about the investigated hyper-parameters: The Xavier method (Glorot and Bengio 2010) is used for weight initialization. The Relu activation function is used due to the best results in terms of accuracy. The Adam Optimizer (Kingma and Ba 2014) was found to be most efficient. A descending size of the feature maps in the encoder network was selected. Average-Pooling is used for sub-sampling in a Gaussian resolution pyramid.

In place of CNN also Fully Connected Neural Networks (FCNN) are used as encoders. The difference between FCN- and CNN based features is the locality of the feature properties in the original curves. CNN are better suited if local properties are more relevant, otherwise FCN are the better choice.

Resistance spot welding sample process

Resistance spot welding is used to connect metal components. In the present application, the components are steel sheets. The steel sheets are squeezed locally by the welding gun electrode tips and form a contact area. The latter is heated by electric current with simultaneously continued electrode force until the sheet material starts melting in the contact area. When the current is shut off, the material re-solidifies again and forms a welding spot, which joins the two (or more) sheets. The resulting welding spot usually has the form of an ellipsoid and is characterized by its maximum equatorial diameter. The basic parameters governing the welding process are (1) the time, for which current flow and electrode force are maintained ('welding time'), (2) the level of the current between the electrodes ('welding current') (3) and the squeezing force exerted on the sheet compound by the electrodes ('electrode force'). Figures 3 and 4 show the principle resistance spot welding arrangement and procedure.

The process data consists of samples of the waveforms (Fig. 5) of electrode voltage (V) and welding current (kA) measured within a given (varying) welding time. These values can be used to calculate the power curve (kW) as shown in Fig. 5.

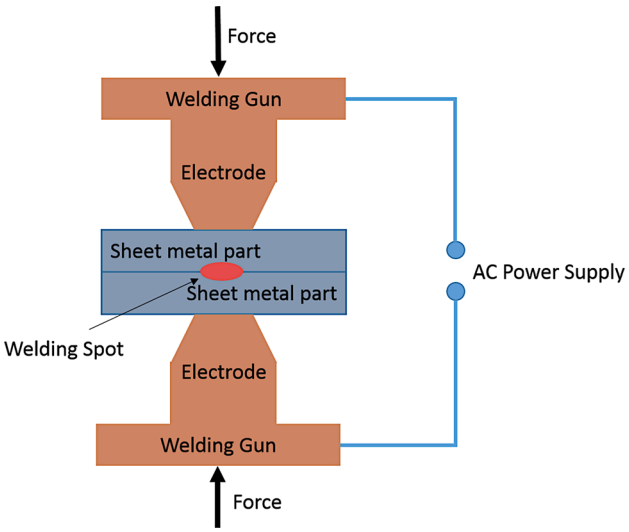

Fig. 3 Resistance spot welding arrangement

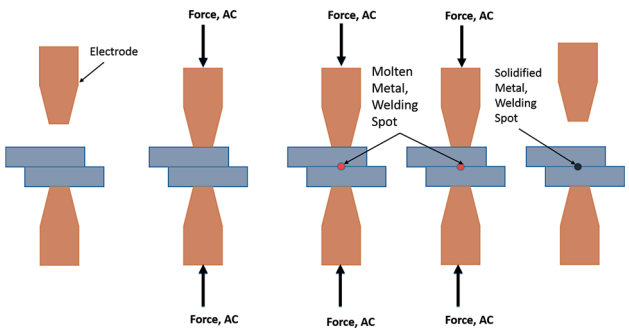

Fig. 4 Resistance spot welding procedure

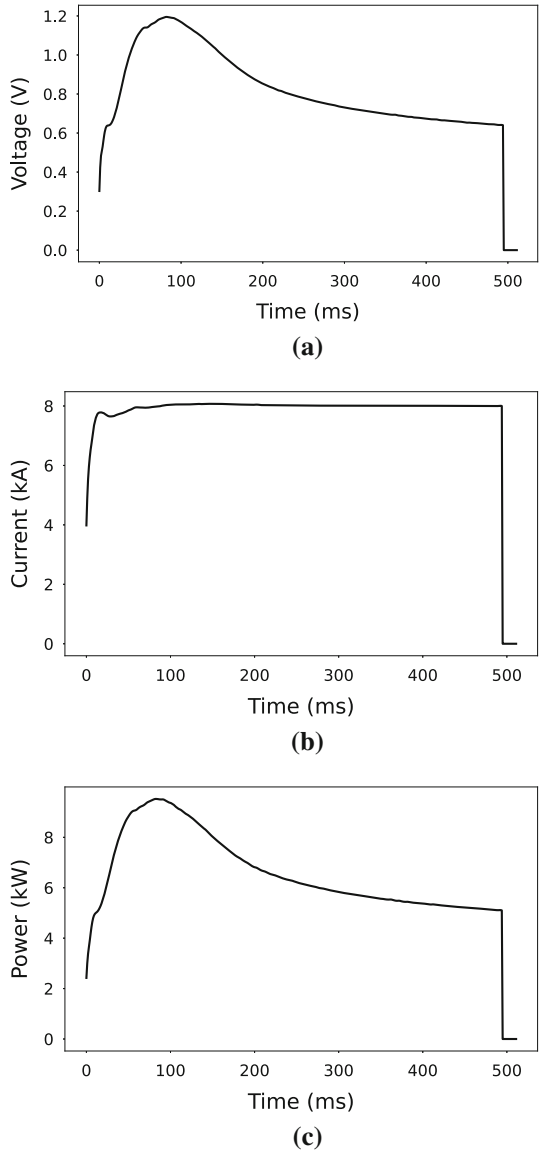

Fig. 5 Voltage, current and power curves of a welding process

Specifics of the resistance spot welding evaluation case

The problem to be solved in the evaluation case is to find universal features describing all possible welding curves independently for all specific conditions. These are the welding gun types (reflecting geometry and type of drive) and steel sheet combinations (determined by materials, thicknesses and number of sheets). All these conditions have influence on the shape of the welding curves, which has to be reflected by the extracted features. The available experimental data comprise almost 4000 power curves with three different welding guns with different geometry/ drive combinations labeled as guns '1', '2'and '3' and eight steel sheet combinations (SSC) of different standard and high-strength steels. The welding process parameters are summarized in Table 2.

Table 2 Welding process parameters

Parameter	Value
Gun types	Gun1, Gun2, Gun3
Material combinations	SSC1, SSC2, …, SSC8
Welding time	Milliseconds (ms)
Welding current level	Kiloamps (kA)
Electrode force	Kilonewtons (kN)

Preprocessing

The following section describes the preprocessing of the sampled raw data.

The considered power curves are sampled at 1 ms intervals and last for 512 ms, where the power may be cut (be zero) after shorter periods. The power curves are determined by the total resistance of the steel sheets plus the resistance of the welding gun itself, which is constant and irrelevant for the welding process. Therefore the resistance R_0 is measured during a "weld" without steel sheets and the corresponding welding gun power contribution subtracted from the calculated power values of the welding curve. ed via a discrete Gaussian filter with a standard deviation of 1.7 ms and a length of the Gaussian window of 6 ms. The curves are standardized to have zero mean and unit variance.

Instantiated encoder topologies and parameters for the spot welding application

The MLP topology (see Fig. 6a and Table 3) includes three fully hidden connected layers that halve the size of the preceding layer. The result of the third hidden layer is mapped to the bottleneck size by a fourth hidden layer, the 'bottleneck layer'. The topology is formulated as a pattern following:

Input → [Fc, Relu] * 4 → Bottleneck

The calculated power curves are subject to measurement noise and smooth

The layout of the CNN topologies (CNN-VA1, CNN-VA2 and CNN-VB1 in Fig. 6 follows the findings of "Time-series analysis of process data" section. The first convolutional layer of each network has a filter kernel size of 25. This allows the detection of local structures of the power curves. Another common property is a stride of length one and thereby no sub-sampling in the convolutional layers. Sub-sampling is performed instead in the pooling layers with stride of length two, halfing the pooling layer input. Each convolutional layer of all topologies contains a Relu activation function to process the convolution result. In each topology, the convolu-

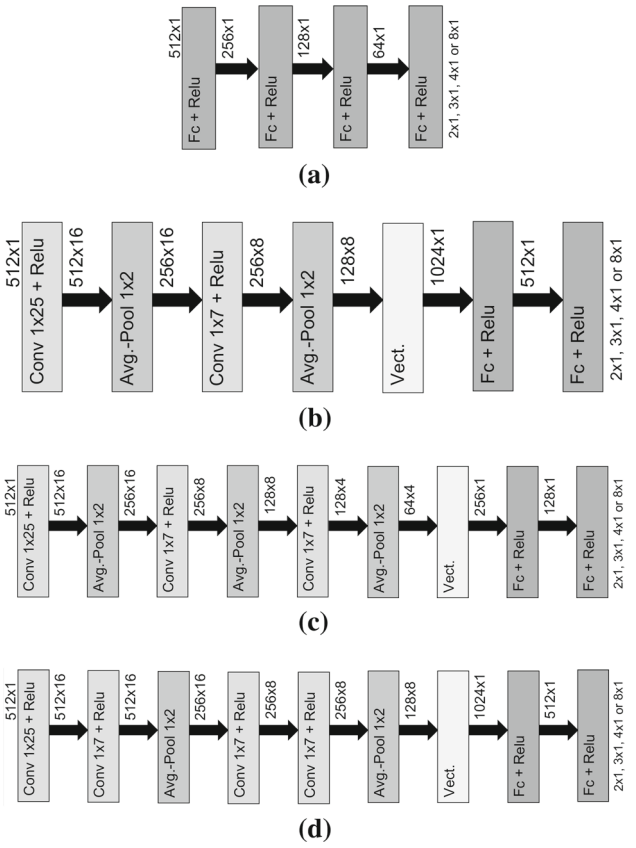

Fig. 6 Instantiated encoder topologies with **a** MLP, **b** CNN VA1, **c** CNN VA2 and **d** CNN VB1

tional layers and the pooling layers are stacked multiple times and their output is vectorized. This high-dimensional vector is finally processed by a combination of two consecutive fully connected layers with Relu activation function to produce the low-dimensional feature vector. The topologies differ in the number of the convolutional and pooling layers (variant A1, variant A2) and in their arrangement (variant B1).

CNN-VA1 consists of two consecutive blocks composed of one convolutional and one pooling layer. The result of the last pooling layer is vectorized and used for further processing by the fully-connected network combination. The formula of CNN-VA1 in pattern notation is:

Input → [Conv, Relu, Pool] * 2 → [Fc, Relu] * 2 → Bottleneck

The Structure CNN-VA2 has one additional block of Conv-Relu and Pooling layer and is otherwise like CNN-VA1. In pattern notation CNN-VA2 is:

Input → [Conv, Relu, Pool] * 3 → [Fc, Relu] * 2 → Bottleneck

The effect of the additional block is twofold: Higher order relations between curve features are considered and the size of the vector layer is halfed due to the additional pooling layer.

Table 3 Instantiated encoder topologies

Model	Layer	Layer type	Output dim.
MLP	0	Input layer	512 × 1
	1	Fc + Relu	256 × 1
	2	Fc + Relu	64 × 1
	3	Fc + Relu	2, 3, 4, 8 × 1
CNN VA1	0	Input layer	512 × 1
	1	Conv 1 × 25 + Relu	512 × 16
	2	Avg.-Pool 1 × 2	256 × 16
	3	Conv 1 × 7 + Relu	256 × 8
	4	Avg.-Pool 1 × 2	128 × 8
	5	Vect.	1024 × 1
	6	Fc + Relu	512 × 1
	7	Fc + Relu	2, 3, 4, 8 × 1
CNN VA2	0	Input layer	512 × 1
	1	Conv 1 × 25 + Relu	512 × 16
	2	Avg.-Pool 1 × 2	256 × 16
	3	Conv 1 × 7 + Relu	256 × 8
	4	Avg.-Pool 1 × 2	128 × 8
	5	Conv 1 × 7 + Relu	128 × 4
	6	Avg.-Pool 1 × 2	64 × 4
	7	Vect.	256 × 1
	8	Fc + Relu	128 × 1
	9	Fc + Relu	2, 3, 4, 8 × 1
CNN VB1	0	Input layer	512 × 1
	1	Conv 1 × 25 + Relu	512 × 16
	2	Conv 1 × 7 + Relu	512 × 16
	3	Avg.-Pool 1 × 2	256 × 16
	4	Conv 1 × 7 + Relu	256 × 8
	5	Conv 1 × 7 + Relu	256 × 8
	6	Avg.-Pool 1 × 2	128 × 8
	7	Vect.	1024 × 1
	8	Fc + Relu	512 × 1
	9	Fc + Relu	2, 3, 4,8 × 1

CNN-VB1 has the same general structure as CNN-VA1, but with different blocks, which consist of two consecutive conv layers, followed by one pooling layer. CNN-VB1 is in pattern notation:

Input → [Conv, Relu, Conv, Relu, Pool] * 2 → [Fc, Relu] * 2 → Bottleneck

The final output of the last fully connected (plus Relu) layer forms the feature vectors in each case. Every architecture is investigated with two, three, four and eight features.

In addition to the investigation of the proposed Neural Network structures, the linear feature extraction via PCA is used to create a performance baseline for the comparison. The number of output components equals two, three, four and eight, analogous to the feature vector size.

Feature-space processing networks architecture and parameters of the spot welding application

The data of the feature vector layer are used as input data for the feature-vector processing networks.

The processing network structures used for the classifications tasks (the three welding gun types and the eight steel sheet combinations), for the estimator task (of the welding spot diameter) and the decoder task (reconstruction of the input welding curve) are shown in Fig. 7. The low-dimensional feature-space input is first non-linearily transformed to a higher-dimensional space to enable the formation of a non-linear class-separation surface of the classifier. This is achieved by a fully connected layer with Relu activation function. This is followed by another such layer, reducing again the dimensionality to deliver the final features for the Softmax classification layer, which assign confidence values between zero and one to the class outputs. The estimator has the same first two processing layers, which feed a linear final regressor, giving a float output value for the welding spot diameter. The structure details can be taken from Fig. 7. The decoder has a typical up-sampling layout by incresing the dimensionality from the feature vector size up to the number of sampling values of the process curve by three fully-connected layers with Relu activation function and a subsequent linear layer: Feature vector → [Fc+Relu]*3 → Fc. The number of neurons can be seen in Fig. 7 and Table 4.

Evaluation of different latent-space feature extractors for resistance spot welding process curves

The approach presented in "Method for the determination of the latent feature space" and "Concept instantiations for resistance spot welding processes" sections is evaluated in two steps. In the first step, for every task, the encoder block is connected only to one single processing task to check if the dimensionality of the feature vector space is in principal sufficient to bear the required information after training the corresponding single task. After this has been proven, the full multi-task learning is performed on the data. The resulting generalized features are evaluated with all connected feature processing networks to check, if they still carry the information about the conditions. Furthermore, the precision of the input reconstruction is analyzed for the generalized features.

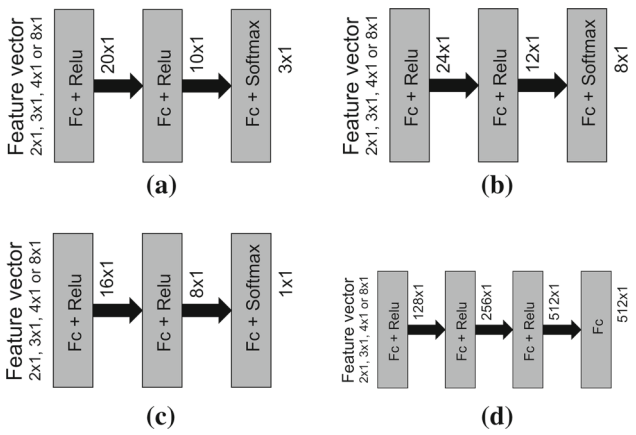

Fig. 7 Feature-space processing networks with **a** classifier welding gun type, **b** classifier steel sheet combination, **c** estimator welding spot diameter and **d** decoder

Table 4 Feature-space processing networks

Task	Layer	Layer type	Output dim.
Classifier welding gun type	0	Input layer	2, 3, 4, 8 × 1
	1	Fc + Relu	20 × 1
	2	Fc + Relu	10 × 1
	3	Fc + Softma ×	3 × 1
Classifier steel sheet combination	0	Input layer	2, 3, 4, 8 × 1
	1	Fc + Relu	24 × 1
	2	Fc + Relu	12 × 1
	3	Fc + Softma ×	8 × 1
Estimator welding spot diameter	0	Input layer	2, 3, 4, 8 × 1
	1	Fc + Relu	16 × 1
	2	Fc + Relu	8 × 1
	3	Fc	1 × 1
Decoder	0	Input layer	2, 3, 4, 8 × 1
	1	Fc + Relu	128 × 1
	2	Fc + Relu	256 × 1
	3	Fc + Relu	512 × 1
	4	Fc	512 × 1

Evaluation metrics

In the following the model performance quality measures of each task are presented.

Classification of the Welding Gun Type and the Steel Sheet Combination: The Softmax layer of the classifier delivers estimates of the Posterior of the classes. The class label with maximum Posterior is assigned to the input data. This results in true positive (TP), true negative (TN), false positive (FP) and false negative (FN) predictions, which are presented by the confusion matrix.

The accuracy gives an insight into how many objects are correctly classified:

$$Accuracy = \frac{TP + TN}{FP + FN + TP + TN} \tag{13}$$

The estimator quality is measured by comparing the estimated diameter d_{est} and the ground truth diameter d_{real} of

all K curves in the sample. The measure is the mean squared difference between prediction and ground truth:

$$MSE = \frac{1}{K}\sum_{i=1}^{K}(d_{real_i} - d_{est_i})^2 \tag{14}$$

The reconstruction error of the decoder is represented by the Mean Relative Absolute Deviation (MRAD). The sampled, measured power curve is represented by $\mathbf{p}_{real}$ and the corresponding, reconstructed power curve by $\mathbf{p}_{rec}$. The MRAD is then the mean of the relative, absolute differences between the $j = 1, \ldots, N$ vector components, averaged over all K samples.

$$MRAD = \frac{1}{K}\sum_{i=1}^{K}\frac{1}{N}\sum_{j=1}^{N}\frac{|p_{real_{j,i}} - p_{rec_{j,i}}|}{p_{real_{j,i}}} \tag{15}$$

Evaluation methods and training

The neural networks use the power curves as input data, which are subject to the preprocessing of "Preprocessing" section. The sample is split in 15% test data and 85% training data. The sample distributions of the different welding gun types and steel sheet combinations are stratified in the data set. In addition, the data for the training is divided into several batches. After each epoch, the entire training and test data set is applied to the network to determine the respective quality measures of "Evaluation metrics" section. Based on these measures, early-stopping (Prechelt 2012) is applied to save the best interim result in order to prevent overfitting. Early stopping should be used almost universally to prevent overfitting (Goodfellow et al. 2016) in Multilayer Perceptrons and Convolutional Neural Networks, why it is used in combination with L2 regularization (Krogh and Hertz 1991) in this work. The training data is stochastically re-sampled within each epoch. The training comprises 4000 epochs.

Each network is trained and evaluated ten times with differently sampled training and test data. From these ten evaluation results, the mean and the standard deviation are calculated to show the quality and robustness of the machine learning results, where a higher standard deviation implies lower robustness of the topology w.r.t. the data.

Evaluation results with generalized features

The results of all tasks (welding gun classification, steel sheet combination classification, estimation of the welding spot diameter, and reconstruction of the welding curve) are shown for all architectural variants (MLP, CNN VA1, CNN VA2, CNN VB1, as declared in "Instantiated encoder topologies and parameters for the spot welding application" section) and for PCA feature extraction (as conventional baseline) in the bottleneck sizes two, three, four and eight. This is followed by a comparison of the implemented architectures based on the quality criteria.

The performance (quality and robustness) of the CNN encoder structures slightly outperform MLP in most cases and both neural structures are significantly superior to the linear PCA baseline. The performance improves with the size of the bottleneck layer for CNN, MLP and PCA. This is shown in Fig. 9, which shows the dependency of the quality metrics for the tasks of welding gun and steel sheet combination classifications, of the estimation of the welding spot diameter and the reconstruction error of the process curves on the bottleneck size N of the topology CNN VA2. The quality metrics of different topologies, all with the same bottlenck size of eight are compared in Fig. 8. The different network topologies show different quality rank order for different tasks. The decision upon a dedicated topology is determined by the average over the weighted ranks of all tasks. The rank of a task is weighted by the relative importance of the respective task. In this work, no task preference is assumed and weight one used for all tasks. The resulting average rank values of all topologies (except the linear PCA) are comparable, while CNN VA2 shows the slightly best overall performance (including robustness), as shown in Figs. 8 and 9. The evaluation shows that even CNN with bottleneck size as small as three has smaller, but sufficient representation power. The performance of the representative top-ranking CNN VA2 structure with bottleneck sizes (denoted as bs) eight and three are compared to PCA in the Table 5.

The evaluation results show that a low-dimensional, generalized feature space can be found via CNN-architectures. Those features do not only allow a sufficient reconstruction of the welding curves, but also bear the information on the process conditions and on the process result. This was proven

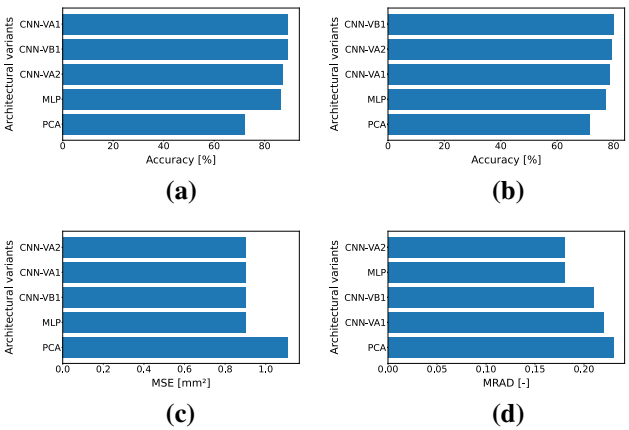

Fig. 8 Comparison of the results for bottleneck size = 8 of the investigated topologies for the tasks **a** welding gun classification, **b** steel sheet combination (ssc) classification, **c** welding spot diameter estimation and **d** reconstruction

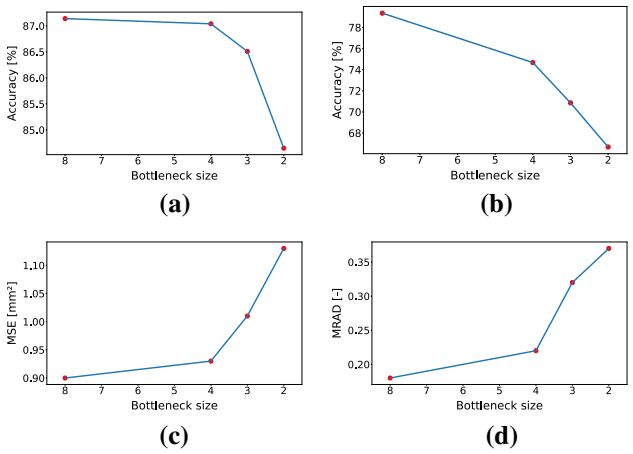

Fig. 9 Comparison of the results for different bottleneck sizes of the top-ranking topology CNN VA2 for the tasks **a** welding gun classification, **b** steel sheet combination (ssc) classification, **c** welding spot diameter estimation and **d** reconstruction

by the classification quality with respect to the process conditions of the welding gun and steel sheet combination and the estimation quality, all of them based on the generalized features. The resulting feature space is not only universal with respect to reconstruction, process conditions and process result, but also low-dimensional, as required.

The training of CNN structures with a given bottleck size, was in some cases switching off some of the bottleneck neurons by constantly setting their outputs to zero. This means that the feature space dimension was effectively smaller than the bottleneck size after training. Some CNN structures with bottleneck size three were setting one bottleneck neuron to zero, thus creating an only two-dimensional latent space, which makes them especially well suited for studies including visualization. Therefore they are used in the subsequent chapters despite their lower performance compared to CNN structures with higher bottleneck sizes.

Latent-space representation and conditional generative and transformation models of resistance spot welding process curves

First, the welding curve representations are compared in the latent space, which results from only an auto-encoder, with a latent-space representation from multi-task learning. Next the conditional densities are estimated in the multi-task-learned latent space from the encoded sample curves via Kernel-density (Rosenblatt 1956) approximation. The density functions of different conditions (two different welding guns) are used to estimate a transformation of the latent space variables, which transforms one density on the other. The rep-

Table 5 Evaluation results

	Accuracy (%)	Standard deviation of accuracy (%)
Welding gun classification		
PCA bs = 8	72.27	2.20
CNN VA2 bs = 8	87.14	1.64
CNN VA2 bs = 3	86.51	1.34
Steel sheet combination (ssc) classification		
PCA bs = 8	71.62	1.89
CNN VA2 bs = 8	79.36	1.74
CNN VA2 bs = 3	70.86	4.95
Welding spot diameter estimation	MSE (mm^2)	Standard deviation of MSE (mm^2)
PCA bs = 8	1.11	0.08
CNN VA2 bs = 8	0.90	0.03
CNN VA2 bs = 3	1.01	0.13
Reconstruction	MRAD	Standard deviation of MRAD
PCA bs = 8	0.23	0.09
CNN VA2 bs = 8	0.18	0.06
CNN VA2 bs = 3	0.32	0.11

The accuracy is defined in Eq. 13. The measured ground truth of the spot diameter in the sample has a range between 2.5 and 8 mm and a standard deviation of 0.6 mm

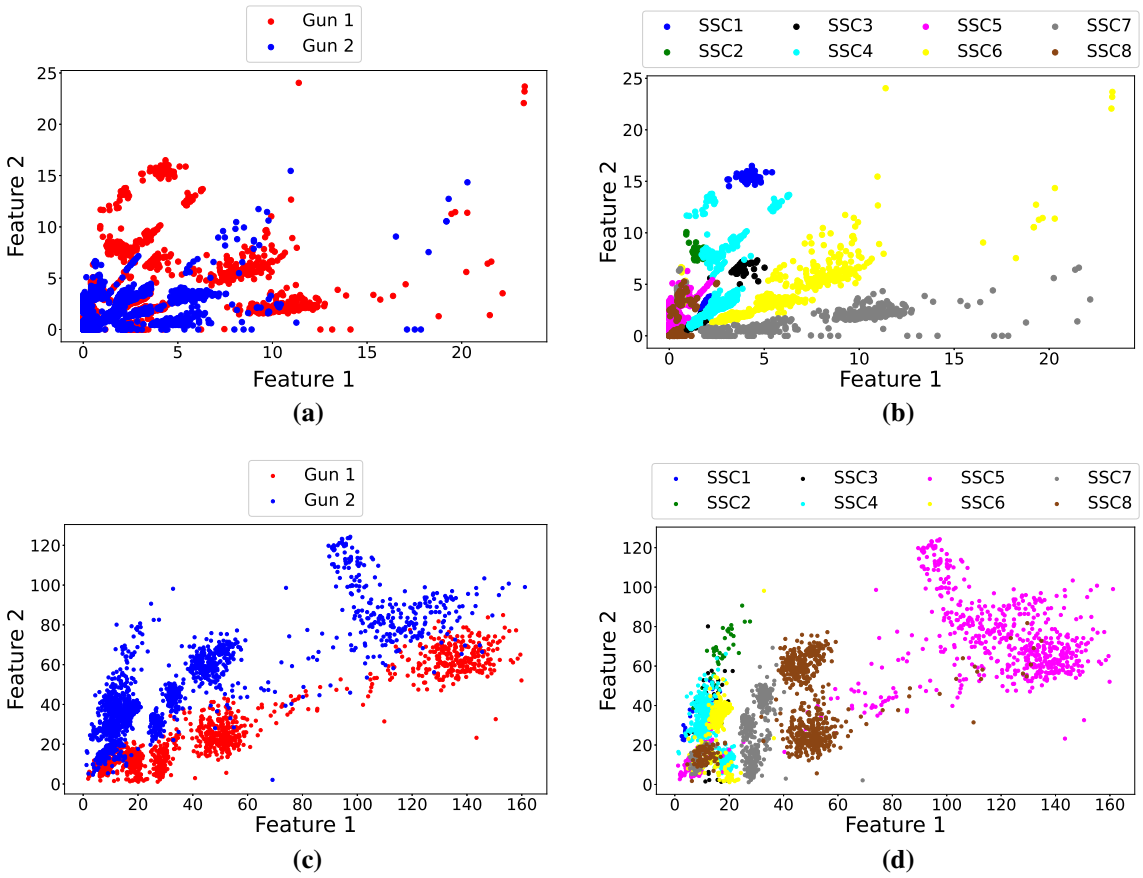

Fig. 10 Comparison of representations in feature spaces learned with autoencoder-only learning (top row) and with multitask learning (bottom row). Left column: welding gun color labeled sample points (red: Gun 1, blue: Gun 2). Right column: SSC color labeled sample points (color code: see legend) (Color figure online)

resentations of individual welding curves from one welding gun are transformed accordingly to the other gun, then reconstructed by the decoder and compared to the original curve belonging to the nearest neighbour representation of the target gun. We restrict our experimental investigations below to the case of two welding guns only, because the third welding gun (which was used in MTL training) has only a low number of samples, preventing a meaningful estimation of a conditional density.

Influence of MTL on latent space structure and densities

Multitask-Learning with auto-encoder, classifier and estimator loss functions has the effect of separating the feature representations of the welding curves according to the respective semantics of the classifiers and estimators. This effect is shown in Fig. 10, where the distributions of the sample curve representations are compared in the latent space, resulting from MTL and autoencoder-only learning. The improvement in arranging the representations in semantically separated areas by MTL is clearly visible for the welding guns (Fig. 10 bottom left) and steel sheet combinations SSC (Fig. 10 bottom right), which are much better separated in the MTL case. From these samples the corresponding conditional densities $p_{est}(\mathbf{x}|c_{gun}, c_{ssc})$, $\mathbf{x} \in$ latent space (conditions: class labels of welding gun type c_{gun} and of steel sheet combination c_{ssc}) can be estimated from the condition-filtered latent-space sample points using e.g. the Kernel density method. The resulting two welding-gun conditional densities for the specific steel sheet combination SSC 8 $p_{est}(\mathbf{x}|c_{gun1}, c_{ssc8})$ and $p_{est}(\mathbf{x}|c_{gun2}, c_{ssc8})$ are shown in Fig. 11 with the sample point clouds, which were used to estimate the densities, in colour blue and red, respectively. The estimated densities represent conditional generative model, which allows the analysis of the effect of the conditions on the densities, which is studied in the next sub-section.

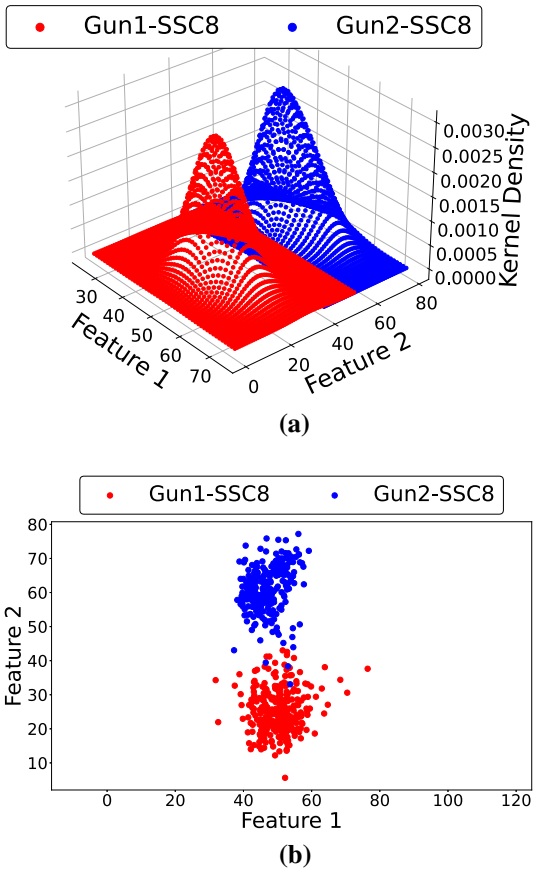

Fig. 11 Estimated density functions (**a**) for Welding Gun 1 and Welding Gun 2, calculated from sample points shown in **b**. For colors see legend (Color figure online)

Transformation models of latent space densities with different conditions

Density functions can be transformed one into the other by a transformation of the latent space variables, which represents the effect of the conditions on the latent space in the sense, that the stochastic generative process of one condition is transformed in a way that it generates samples, which obey the same distribution as for the other condition. In other words: The stochastic generative process under a certain condition is transformed into a stochastic generative process under a different condition. Models of such transformations could be estimated assuming a generic transformation (such as affine transformation) and estimating the transformation parameter values. From multiple such models (parameter value instances) a model of the dependency of the parameters on the conditions can be established, representing a process hyper-model, as introduced in "Introduction" section. Similar ideas are applied in applications of VAEs (as discussed in "Variational and conditional variational autoencoders (VAEs and CVAEs)" section, where -instead of modelling condition-dependency of the densities- only linear interpolations between samples are applied directly in the latent space. The remaining part of the subsection shows the estimation of the latent-space transformation, which maps the density $p_{est}(\mathbf{x}|c_{gun1}, c_{ssc8})$ on density $p_{est}(\mathbf{x}|c_{gun2}, c_{ssc8})$ as an example of this procedure. We use a linear transformation matrix $\mathbf{A}$ of the latent-space coordinates $\tilde{\mathbf{x}}' = \mathbf{A}\tilde{\mathbf{x}}$ in homogeneous coordinates (to also include shift transformations) $\tilde{\mathbf{x}} = [\mathbf{x}, 1]^T$ and minimze the squared difference loss function to yield the estimated transformation

$$\mathbf{A}^* = argmin_{\mathbf{A}} \left[p_{est}(\mathbf{A}\tilde{\mathbf{x}}|c_{gun1}, c_{ssc8}) - p_{est}(\tilde{\mathbf{x}}|c_{gun2}, c_{ssc8})\right]^2. \tag{16}$$

The minimization is performed by equidistantly sampling the (via Kernel-density) estimated densities in the original and transformed space and finding the optimal transformation. The optimization is performed with an ADAM optimizer (Kingma and Ba 2014). A regularization term is included in the loss function, which keeps the integral of the transformed density close to one, taking care of the normalization property also of the transformed density. This procedure is applied to the two densities as introduced above to reveal the influence of the welding gun in the case of ssc8. The two original densities with fixed ssc8 condition and Gun 1 and Gun 2 conditions, estimated from the corresponding samples, are shown with the overlayed transformed Gun 1 density in Fig. 12. It can be seen that the chosen linear transform is sufficient in this case and the transformed Gun 1 density is in good agreement with the Gun 2 density.

If the transformation is applied directly to the Gun 1 sample points, we also get a good agreement with the point cloud of Gun 2 as can be seen in Fig. 13. To see how a change in process conditions (such as tool types, workpiece material properties, and so on) changes the evolution of the process (as reflected by the process curve), we estimate a process curve transformation depending on the condition change. The result in Fig. 13 shows that the point cloud of the measured sample points under condition 1 almost coincides with the tranformed point cloud of the sample points under condition 2, where each point was mapped by the transformation matrix, which was optimized to map the density function under condition 2 on the density function under condition 1. This nearly coincidence implies that a model for the transformation between the generating processes under both conditions was found. Going back to the transformed densities, all quantities, which can be derived from a density function can be transformed as well. This could be the average process curve, which is frequently used in quality measurements, but also the variance of process curves, which is frequently used as a bound for process controls. Any higher-order statistical moment, which has importance for control or quality assessment can also be derived.

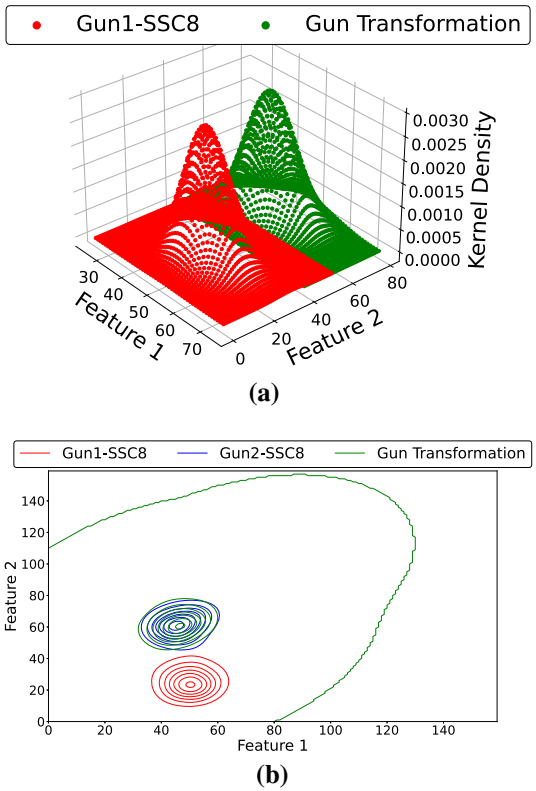

Fig. 12 **a** Transformed density (green color) from estimated linear latent space transformation of Gun 1 density (red color). **b** Contour plot of transformed densities as shown in **a** along with Gun 2 density (blue color) for comparison (Color figure online)

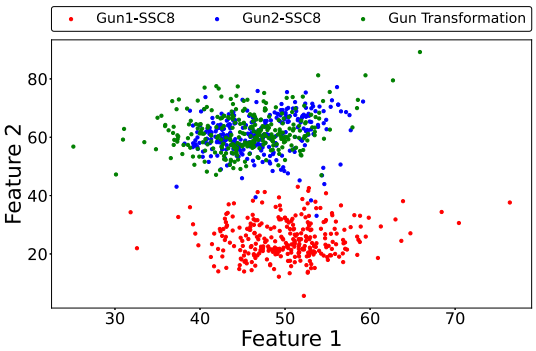

Fig. 13 Result (green) of the feature space transformation on the sample points of Gun 1 (red), overlayed with Gun 2 sample points (blue) (Color figure online)

The comparison of the re-constructed curves of two transformed Gun 1 sample points with the reconstruction of its nearest Gun 2 neighbours in Fig. 14 shows a high similarity.

Additionally, we investigated three further transformations, exploring the condition space of welding guns and steel sheet combinations of the sample application. The results are depicted in Figs. 15, 16 and 17 in Appendix A, with (a) the contour plots of the original densities (red and blue) and of the transformed density (green), (b) the resulting sample

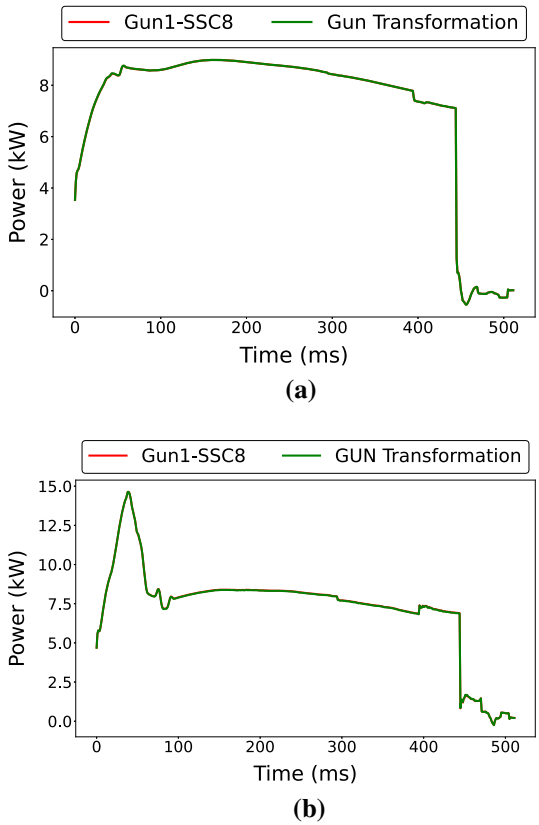

Fig. 14 Reconstructed curves

points (green) after the corresponding feature space transformation, along with the original sample points, and (c) and (d) compare the reconstructed curves of two sample points, the first curve (red) reconstructed from a point drawn from the original sample under condition 1 and the second curve (green) reconstructed from the next transformed neighbour under condition 2 under the previously estimated transformation between condition 1 and condition 2. It can be seen that the previous findings for the condition transformation hold in general true for the sample application.

An in-depth investigation in hyper-modelling, generalizing the inluence of conditions on the processes, which is revealed with the above methods, will be subject of future work.

Conclusion

A method for the extraction of a latent variable space from process curves by means of CNNs has been developed, which uses multi-task learning to let the latent space also reflect dependencies of process conditions. It is built by attaching a decoder (reconstructing the process curve), a classifier (for

discrete conditions) and an estimator (for continuous conditions) to the output layer of the encoding CNN. The result is a latent space at the CNN output (its bottleneck layer), where the process curves are represented in a way, which is also sensitive to the conditions. The validity of the concept was proven with resistance spot welding as application example. Different CNN and FCNN architectures were explored and compared. An only three-dimensional latent space was found, which enables the welding curve reconstruction, the classification of the welding gun in use and the welded steel sheet combination, as well as the estimation of the welding spot diameter independently and with sufficient accuracy. A comparison with a latent space made up of three PCA features showed the superiority of the non-linear feature extraction with CNN and FCNN, where using the CNN showed a slight advantage over using FCNN.

The latent space resulting from the proposed approach enables the transformation of the stochastic processes (their probability denisities), which generate the process curves instances, to other process conditions. This is an important pre-condition for hyper modeling of the processes, where the dependencies of the process on varying conditions is captured and can be used to predict the properties of previously unseen processes. Investigations of the proposed approach for hyper modeling will be the subject of future work. As a by-product, conditions can be easily found from a process curve, and hidden conditions can be detected.

Acknowledgements The authors would like to thank the German Federal Ministry of Education and Research (BMBF) for funding the presented research under Grant #03FH061PX5.

Funding Open Access funding enabled and organized by Projekt DEAL

Appendix A: Results of further transformations

See Figs. 15, 16 and 17.

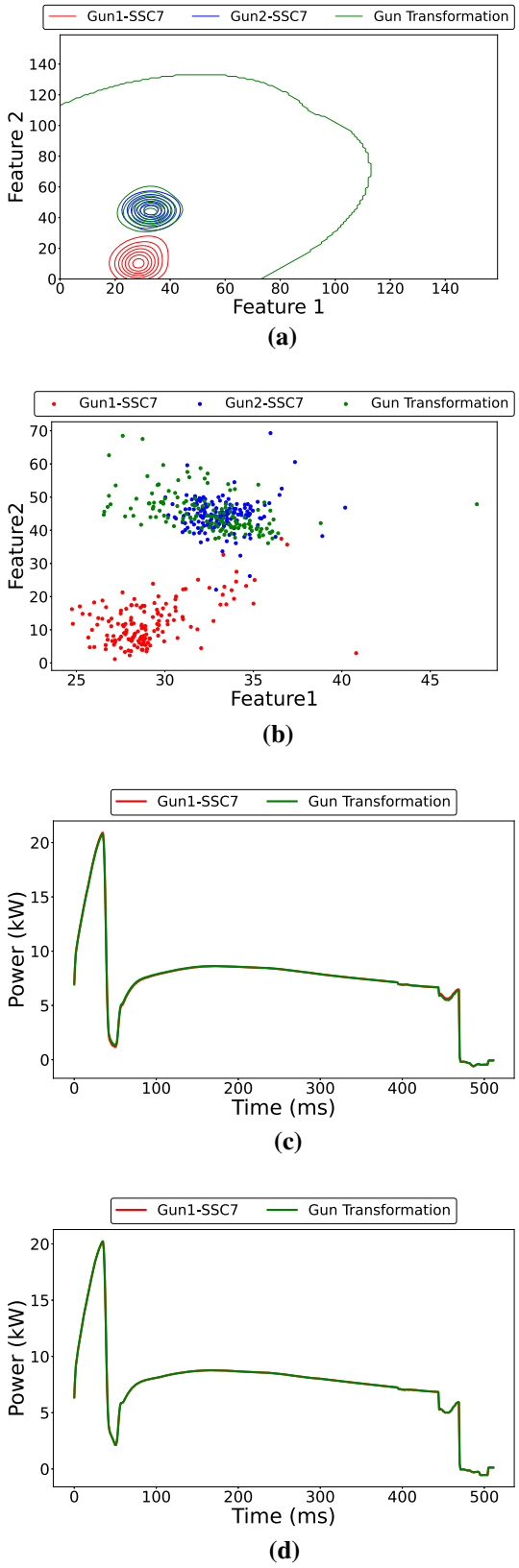

Fig. 15 Investigated transformation: (Gun1, SSC7) → (Gun2, SSC7)

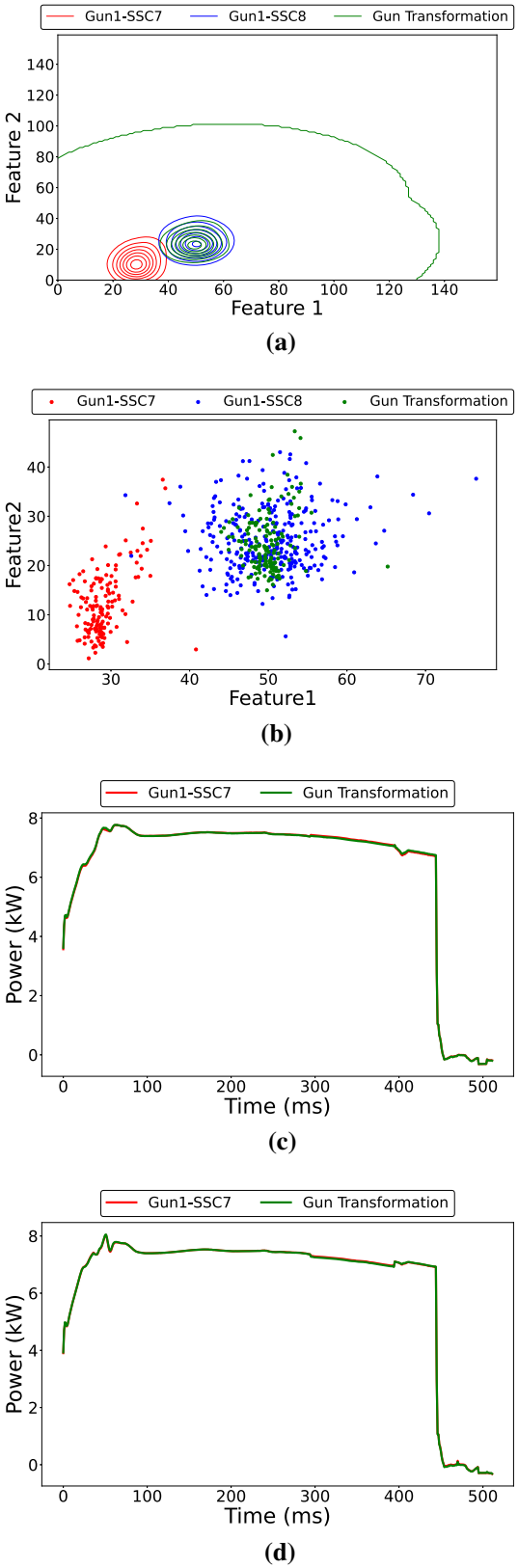

Fig. 16 Investigated transformation: (Gun1, SSC7) → (Gun1, SSC8)

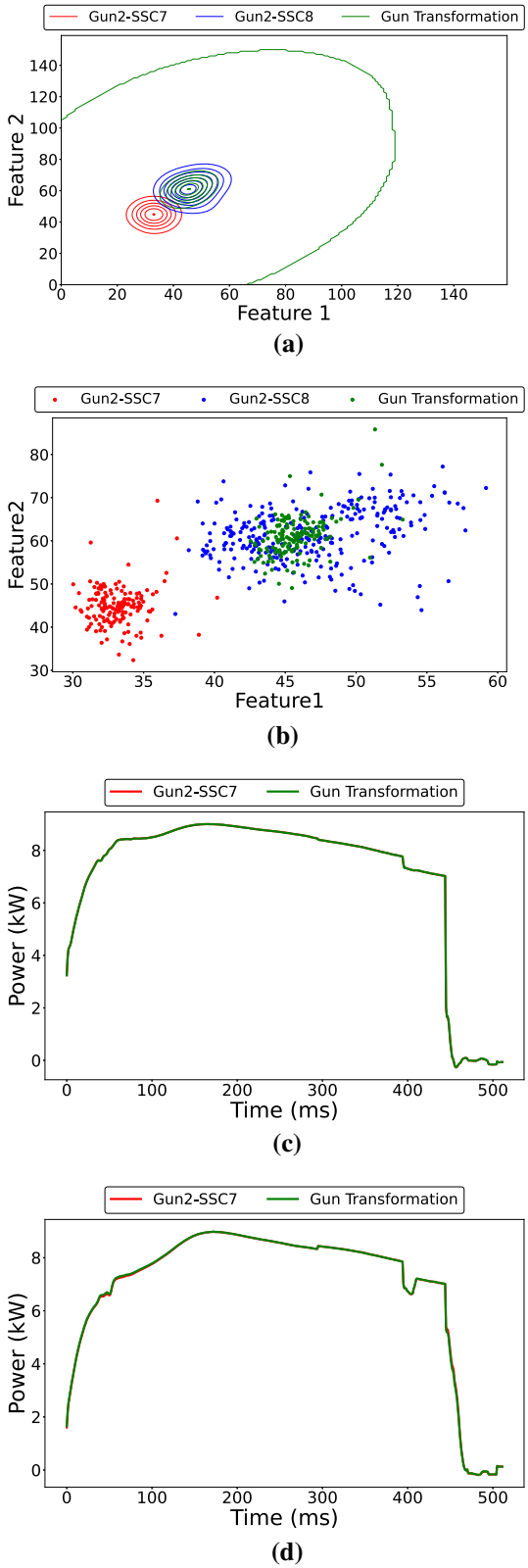

Fig. 17 Investigated transformation: (Gun2, SSC7) → (Gun2, SSC8)

References

Baydogan, M. G., Runger, G., & Tuv, E. (2013). A bag-of-features framework to classify time series. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 35,* 2796–2802.

Belkin, M., & Niyogi, P. (2001). Laplacian Eigenmaps and spectral techniques for embedding and clustering. In *Proceedings of the 14th international conference on neural information processing systems: Natural and synthetic*, NIPS'01, (Cambridge, MA, USA) (pp. 585–591). MIT Press.

Berthelot, D., Raffel, C., Roy, A., & Goodfellow, I. (2019). Understanding and improving interpolation in autoencoders via an adversarial regularizer. In *International conference on learning representations*.

Caruana, R. (1997). Multitask learning. *Machine Learning, 28,* 41–75.

Chen, Y., Keogh, E., Hu, B., Begum, N., Bagnall, A., Mueen, A., & Batista, G. (2015). The UCR time series classification archive. www.cs.ucr.edu/~eamonn/time_series_data/.

Cox, M. A. A., & Cox, T. F. (2008). *Multidimensional scaling* (pp. 315–347). Berlin, Heidelberg: Springer.

Dai, W., Dai, C., Qu, S., Li, J., & Das, S. (2017). Very deep convolutional neural networks for raw waveforms. In *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)* (pp. 421–425).

Esteban, C., Hyland, S. L., & Rätsch, G. (2017). Real-valued (medical) time series generation with recurrent conditional gans.

Fischer, S., Hensgen, O., Elshaabiny, M., & Link, N. (2015). Generating low-dimensional, nonlinear process representations by ordered features (Vol. 48, 05).

Gao, X., & Liang, J. (2013). Manifold learning algorithm DC-ISOMAP of data lying on the well-separated multi-manifold with same intrinsic dimension. *Jisuanji Yanjiu yu Fazhan/Computer Research and Development, 50,* 1690–1699.

Ghahramani, Z., & Jordan, M. I. (1997). Factorial hidden Markov models. *Machine Learning, 29,* 245–273.

Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In Teh, Y. W., Titterington, M. (Eds.), *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, vol. 9 of *Proceedings of Machine Learning Research*, (Chia Laguna Resort, Sardinia, Italy), pp. 249–256, PMLR, 13–15 May.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning* (p. 426). MIT Press. http://www.deeplearningbook.org.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., et al. (2014). Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems* (Vol. 27, pp. 2672–2680). Red Hook: Curran Associates, Inc.

Graves, A. (2013). Generating sequences with recurrent neural networks. *CoRR*, vol. abs/1308.0850.

Hou, X., Shen, L., Sun, K., & Qiu, G. (2016). Deep feature consistent variational autoencoder. *CoRR*. vol. abs/1610.00291.

Keogh, E., & Ratanamahatana, C. A. (2005). Exact indexing of dynamic time warping. *Knowledge and Information Systems, 7,* 358–386.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *CoRR*, vol. abs/1412.6980.

Kohonen, T., Schroeder, M. R., & Huang, T. S. (Eds.) (2001). *Self-Organizing Maps*, 3rd edn. Berlin, Heidelberg: Springer.

Krogh, A., & Hertz, J. A. (1991). A simple weight decay can improve generalization. *Advances in Neural Information Processing Systems, 4,* 950–957.

Larochelle, H., Erhan, D., & Bengio, Y. (2008). Zero-data learning of new tasks. In *Proceedings of the 23rd national conference on artificial intelligence—volume 2*, AAAI'08 (pp. 646–651). AAAI Press.

Larson, R., & Odoni, A. (1981). *Urban operations research*, ch. 7, simulations. Prentice-Hall.

Lin, J., Keogh, E., Wei, L., & Lonardi, S. (2007). Experiencing sax: A novel symbolic representation of time series. *Data Mining and Knowledge Discovery, 15,* 107–144.

Link, N., Pollak, J., & Sarveniazi, A. (2016). Concept for finding process models for new classes of industrial production processes.

Merz, C. J., & Pazzani, M. J. (1999). A principal components approach to combining regression estimates. *Machine Learning, 36,* 9–32.

Prechelt, L.(2012). *Early stopping—But when?* (pp. 53–67). Berlin: Springer.

Rakthanmanon, T., Campana, B., Mueen, A., Batista, G., Westover, B., Zhu, Q., Zakaria, J., & Keogh, E. (2012). Searching and mining trillions of time series subsequences under dynamic time warping. In *Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining*, KDD '12 (New York, NY, USA) (pp. 262–270). ACM.

Reis, J., Gonçalves, G., & Link, N. (2017). Meta-process modeling methodology for process model generation in intelligent manufacturing. In *IECON 2017—43rd annual conference of the IEEE industrial electronics society* (pp. 3396–3402).

Rosenblatt, M. (1956). Remarks on some nonparametric estimates of a density function. In *The Annals of Mathematical Statistics* (pp. 832–837).

Rosipal, R., & Trejo, L. J. (2002). Kernel partial least squares regression in reproducing kernel Hilbert space. *Journal of Machine Learning Research, 2,* 97–123.

Roweis, S. T., & Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science, 290*(5500), 2323–2326.

Schölkopf, B., Smola, A., & Müller, K.-R. (1997). Kernel principal component analysis. In W. Gerstner, A. Germond, M. Hasler, & J.-D. Nicoud (Eds.), *Artificial neural networks–ICANN'97* (pp. 583–588). Berlin: Springer.

Senn, M. (2013). *Optimale Prozessfuehrung mit merkmalsbasierter Zustandsverfolgung*, vol. XVI, 222 pp. KIT. Scientific Publishing.

Socher, R., Ganjoo, M., Manning, C. D., & Ng, A. Y. (2013). Zero-shot learning through cross-modal transfer. In *Proceedings of the 26th international conference on neural information processing systems - volume 1*, NIPS'13, (USA) (pp. 935–943). Curran Associates Inc.

Sohn, K., Lee, H., & Yan, X. (2015). Learning structured output representation using deep conditional generative models. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, & R. Garnett (Eds.), *Advances in neural information processing systems*. Red Hook: Curran Associates, Inc.

Souvenir, R., & Pless, R. (2005). Manifold clustering. In *Tenth IEEE international conference on computer vision (ICCV'05)* (Vol. 1, pp. 648–653).

Tenenbaum, J. B., de Silva, V., & Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science, 290*(5500), 2319–2323.

Vinzi, V. E., Chin, W. W., Henseler, J., & Wang, H. (2010). *Handbook of partial least squares: Concepts, methods and applications* (1st ed.). Berlin: Springer.

Wang, Z., Yan, W., & Oates, T.(2017). Time series classification from scratch with deep neural networks: A strong baseline. In *2017 International joint conference on neural networks (IJCNN)* (pp. 1578–1585).

Yoon, J., Jarrett, D., & van der Schaar, M. (2019). Time-series generative adversarial networks. In *Advances in neural information processing systems* (Vol. 32, pp. 5508–5518). Curran Associates, Inc.