Knowledge transfer across cell lines using Hybrid Gaussian Process models with entity embedding vectors

Clemens Hutter[1,2]

Moritz von Stosch[1]

Mariano Nicolas Cruz Bournazou[1,3]

Alessandro Butté[1†]

[1] DataHow AG, Zurich, Switzerland

[2] Chair for Mathematical Information Science, ETH Zurich

[3] KIWI-biolab, Bioprocess Engineering, TU Berlin

[†] Corresponding author

Address c/o ETH Zurich, HCI F137
Vladimir-Prelog-Weg 1
CH-8093 Zurich, Switzerland

Phone +41 (0)44 633 38 98

E-mail a.butte@datahow.ch

Running Title: **Knowledge transfer across cell lines**

Abstract

To date, a large number of experiments are performed to develop a biochemical process. The generated data is used only once, to take decisions for development. Could we exploit data of already developed processes to make predictions for a novel process, we could significantly reduce the number of experiments needed. Processes for different products exhibit differences in behaviour, typically only a subset behave similar. Therefore, effective learning on multiple product spanning process data requires a sensible representation of the product identity. We propose to represent the product identity (a categorical feature) by embedding vectors that serve as input to a Gaussian Process regression model. We demonstrate how the embedding vectors can be learned from process data and show that they capture an interpretable notion of product similarity. The improvement in performance is compared to traditional one-hot encoding on a simulated cross product learning task. All in all, the proposed method could render possible significant reductions in wet-lab experiments.

Keywords: Gaussian Process Regression, Embedding Vector, Transversal Data Analysis, Hybrid semi-parametric modeling, bioprocess development, cell culture

1 Introduction

The production of therapeutic proteins and vaccine at scale requires manufacturing processes with a very precise control of the final drug quality, that are economically viable and are outstanding in the protection of environment, health and safety (Lee et al., 2015; Yu & Kopcha, 2017). The development of these processes is extremely time consuming, highly unreliable and costly, since the process design and conditions need to be tailored to the specific characteristics of the product sought to be produced (Portela et al., 2020). Even for "platform processes", a large number of process design parameters need to be optimized to increase yields and meet product quality specifications (F. Li, Vijayasankaran, Shen, Kiss, & Amanullah, 2010). Recent years have seen a rise of miniaturized high-throughput platforms (fueled by advances in liquid handling stations) that are representative of some part of the production process, enabling parallel cost- and time-efficient experimentation (Rameez, Mostafa, Miller, & Shukla, 2014; Velugula-Yellela et al., 2018; Shrirao et al., 2018; Bushnell et al., 2020).

With the rise of high-throughput technologies (Hans et al., 2020), the process development bottleneck has shifted to i) gathering and analysing the generated experimental data; as well as ii) designing the many parallel experimental runs, such that the data of each run is informative. While statistical Design of Experiment methods have come some way towards addressing this challenge (Politis, Colombo, Colombo, & Rekkas, 2017), these methods largely disregard prior knowledge about the process at hand (von Stosch, Davy, et al., 2014) (in particular when it comes to changes in the cell line), because i) this knowledge is not available in a format that could be explored for experiment design, and ii) though data are produced for similar processes, set-ups for iterative and successive learning (knowledge gathering) from the data are scarce. Hence, horizontal knowledge transfer from product to product, cell line to cell line, remains limited and entire process design spaces need to be studied *de novo* for every product. This implies that for every novel product a similar number of experiments will be required to understand the process behaviour. In previous articles, the possibility of creating more powerful process models has been explored (Narayanan, Sokolov, Morbidelli, & Butté, 2019; Narayanan, Luna, et al., 2020), which can be combined with the use of model-based Design of Experiment (Stosch, 2018; Teixeira, Alves, Alves, Carrondo, & Oliveira, 2006; Ferreira et al., 2014; Brendel & Marquardt, 2008; Kuchemüller, Pörtner, & Möller, 2020; Abt et al., 2018; Anane et al., 2019; Hans et al., 2020) to reduce the experimental effort. However, for a successive acceleration of process development activities novel methods are needed that allow to derive information from a joint analysis of process data generated for different products and further are even capable to predict process behaviour for novel products, from the moment that a limited amount of data has been generated.

To date three approaches for the analysis of multiple product spanning process data are available, as discussed below: one hot encoding (dummy or categorical variables), product specific scaling, or integrating "handcrafted" features that describe the characteristics of the product (e.g. molecular descriptors).

- *One hot encoding*: The product identity for each data point is handled as a categorical process variable and for analysis purposes transformed into a one hot (dummy) variable (Gori, 2018). While for analysis purposes it might be efficient to differentiate between data of different products, the learnings that can be transferred from one product to the next remain limited, since the one hot encoding does not carry information about the similarity of the different products. Furthermore, it is not possible to filter and extract only the information that is useful for the novel product.

- *Scaling*: Scaling can improve the comparability of process data from different products (X. Li et al., 2017; Liu et al., 2019). One can use, for instance, in process scaling to "normalise" for differences in absolute titers or to understand the impact of pH-variations around different set-points. However, if the scaling does not result in making the data more comparable then a joint analysis will likely provide as much insight as an analysis of the data of each product alone.

- *Handcrafted, knowledge-based feature creation*: Integrating features into the analysis that describe the product characteristics can enable cross-product data analysis. To this end, molecular descriptors have been generated and used, giving rise to quantitative structure–activity relationship (QSAR) models, quantitative sequence-activity modelling (QSAM) or convolution graph networks (Schweidtmann et al., 2020). Karlberg et al (Karlberg, von Stosch, & Glassey, 2018) proposed to use mAb structure characteristics for a streamlined implementation of the Quality by Design (QbD) paradigm. However, it is not straightforward to capture the characteristics of a drug substance and additionally select the features of interest from the multitude of generated features for a limited amount of data is also not trivial.

In this work we demonstrate for the first time the potential of product embeddings with Gaussian Process Regression for multiple product spanning process data modeling in bioprocess development. The concept of embedding categorical entities (products in our case) was inspired by the representation of words with high dimensional vectors in the field of natural language processing. These embeddings have many useful characteristics like locating words with similar meaning close by (Pennington, Socher, & Manning, 2014). They can even exhibit *additive compositionality* such that the embedding of the word "Berlin" is closest to the sum of the embedding vectors for "capital" and "Germany" (Mikolov, Sutskever, Chen, Corrado, & Dean, 2013). Usually, word embeddings are learned as weights of a neural network on a generic task (Mikolov, Yih, & Zweig, 2013). In this article we aim to generalize this powerful concept from neural network learning such that it can also be used with Gaussian Process (GP) regression. For bioprocess modeling in particular, GP regression has notable advantages over neural network learning such as lower data requirements and more importantly the possibility to assess the uncertainty of predictions. We hence consider it worthwhile to extend the applicability of entity embedding vectors to this regression algorithm.

Using the embeddings, we seek to explain the similarities in the biological system across products. Variations sourcing from process operations are explicitly described using material balances, see e.g. (Richelle, Lee, Portela, Raley, & Stosch, 2020) for more details. The resulting system of equations naturally constitutes a hybrid semi-parametric model.

In what follows, we first describe the proposed approach to learn product embeddings with Gaussian Process models integrated into a hybrid semi-parametric process model. We derive mathematical insight into the shortcomings of the traditional product representation with one-hot vectors in Gaussian Process regression and show how our learned embeddings provide insight into similarities between products. We then rigorously evaluate the proposed approach using a simulation case study and finalize by summarizing the findings.

2 Material and Methods

2.1 Bioprocess Data in Process development

In the development of a process the aim is to find process conditions (e.g. temperatures, initial concentrations, etc.) that consistently produce a large amount of high quality titer (product concentration) in a short amount of time. To effectively support the development of a process for a novel product, we want a model that can predict the system behaviour for an unseen process condition $\mathcal{E}'$, such that we can use the model for process optimization. The behavior of the system is here represented by a matrix of concentrations $\mathcal{M}' \in \mathbb{R}^{N_{time} \times N_Q}$ for N_Q relevant quantities — here $N_Q = 6$: Viable Cell Density (VCD), Glucose, Glutamine, Ammonia, Lactate and Titer — at N_{time} time points — here $N_{time} = 14$ — throughout the process evolution.

To create such a model, data from $N_{\mathcal{E}}$ experimental runs which had been generated in the past for N_{prods} different products could be used, as we show in the following. For each run $i \in \{1 \ldots N_{\mathcal{E}}\}$, the process condition $\mathcal{E}^i$ and the product $p^i \in \{1, \ldots, N_{prods}\}$ as well as the matrix of concentrations $\mathcal{M}^i \in \mathbb{R}^{N_{time} \times N_Q}$ are considered to be known.

2.2 Hybrid Regression Models

The modeling of the dynamic evolution of the concentrations is inspired by the dynamic material balances of an ideally mixed reactor. Specifically, by discretizing in time, we approximate for each experiment in the training set $i \in \{1, \ldots, N_{\mathcal{E}}\}$ the slope of the change in concentrations between each consecutive pair of measurements as follows

$$y_t^i := \frac{\mathcal{M}_{t+1}^i - \mathcal{M}_t^i}{\Delta t} \tag{1}$$

where $\mathcal{M}_t^i \in \mathbb{R}^{N_Q}$ is the vector of measurements at time t and Δt is the time between two measurements. Further, we construct feature vectors x_t^i by concatenating the concentrations at the beginning of the step $\mathcal{M}_t^i$ with additional features $ExtraFeatures(\mathcal{E}^i) \in \mathbb{R}^k$ such as reactor temperature and pH which are obtained from the experimental design. Doing this for all experiments in the training set we obtain matrices $X \in \mathbb{R}^{(N_{\mathcal{E}} \cdot N_{time}) \times (N_Q + k)}$ and $Y \in \mathbb{R}^{(N_{\mathcal{E}} \cdot N_{time}) \times N_Q}$. We use these to train an arbitrary regression model $\Phi : \mathbb{R}^{N_Q + k} \to \mathbb{R}^{N_Q}$, which describes the reaction kinetics in the reactor analogy.

The prediction for an unseen experimental condition $\mathcal{E}'$ is given by

$$\mathcal{M}_0' = InitialConcentration(\mathcal{E}') \tag{2}$$

$$\mathcal{M}_{t+1}' = \mathcal{M}_t' + \Phi\left(\left(\mathcal{M}_t' \quad ExtraFeatures(\mathcal{E}')\right)\right) \cdot \Delta t \tag{3}$$

Additionally, in case of feeding, the effects of fed masses are accounted for explicitly in the mechanistic mass balances, see e.g. (Richelle et al., 2020). Specifically, equation (3) is then extended to

$$\mathcal{M}_{t+1}' \cdot V_{t+1} = \left[\mathcal{M}_t' + \Phi\left(\left(\mathcal{M}_t' \quad ExtraFeatures(\mathcal{E}')\right)\right) \cdot \Delta t\right] \cdot V_t + \Delta F_{t+1}, \tag{4}$$

where $\Delta F_{t+1} \in \mathbb{R}^{N_Q}$ is the vector of changes in mass due to feeding at time $t+1$ and $V_t \in \mathbb{R}^+$ is the volume at discrete timepoints. The regression target (1) to be approximated with the trained model Φ then reads

$$y_t = \frac{\mathcal{M}_{t+1} \cdot V_{t+1} - \mathcal{M}_t \cdot V_t - \Delta F_{t+1}}{V_t \cdot \Delta t}. \tag{5}$$

In our experiments we have Glucose and Glutamine feeds and hence the vector ΔF contains nonzero entries only in the positions corresponding to those two quantities. All components of the modelled system together constitute a hybrid semi-parametric model, in line with the definition by (von Stosch, Oliveira, Peres, & Feyo de Azevedo, 2014).

2.3 Gaussian Process Regression

Gaussian Process regression (GP) is a particular choice for the class of regression model used in Section 2.2. It approximates a function $f : \mathbb{R}^d \to \mathbb{R}$ where d is the number of features. In this brief outline we assume for simplicity that there is no measurement noise and refer to (Rasmussen & Williams, 2006) for more details. A GP regression model requires a kernel or covariance function $k(\cdot, \cdot) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ which should capture a notion of similarity that is appropriate for the application (Rasmussen & Williams, 2006, Chapter 4). Using the kernel, a prior over the function $f(\cdot)$ is defined. The prior specifies that the function values $\overline{Y} \in \mathbb{R}^{n+1}$ evaluated at a set of $n+1$ points $\overline{X} \in \mathbb{R}^{(n+1)\times d}$ follow a normal distribution

$$\overline{Y} \sim \mathcal{N}(0, K(\overline{X}, \overline{X})), \tag{6}$$

where the matrix $K(\overline{X}, \overline{X}) \in \mathbb{R}^{(n+1)\times(n+1)}$ is given by applying the kernel to each pair of rows: $K(\overline{X}, \overline{X})_{ij} = k(\overline{x}_i, \overline{x}_j)$. We can treat n of these datapoints as observed (i.e. training data X, Y) and the remaining point as a query point for which only x^* is known. A predictive distribution over $f(x^*) = y^*$ is then obtained by conditioning the prior with the observed training data (Rasmussen & Williams, 2006, Section 2.2):

$$\begin{aligned} y^* \mid x^*, X, Y \sim \mathcal{N}(&K(x^*, X)K(X, X)^{-1}Y, \\ &k(x^*, x^*) - K(x^*, X)K(X, X)^{-1}K(X, x^*)) \end{aligned} \tag{7}$$

Among the most widely used kernels is the squared exponential kernel (or radial basis function) with automatic relevance determination (ARD)

$$RBF_\theta(x, x') = exp\left(-\sum_{i=1}^{d} \frac{(x_i - x'_i)^2}{2\theta_i^2}\right), \tag{8}$$

where $\theta \in \mathbb{R}^d$ is a vector of hyper-parameters. For the RBF kernel these parameters are usually called length-scales. When using a parameterised kernel, the prior (6) also depends on θ. The value of θ can be chosen by maximising the likelihood of the observed training data under the prior:

$$\theta := \arg\max_{\theta'} logP(Y|X, \theta') \tag{9}$$

See (Rasmussen & Williams, 2006, Section 5.4.1) for more details.

The one dimensional regression model can easily be extended to functions $\mathbb{R}^d \to \mathbb{R}^t$ by using t independent single output GPs that share hyper-parameters θ. Prediction is then effected (for fixed θ) by using equation (7) t times independently on each of the t targets. To find θ one can maximise the joint objective $\sum_{i=1}^{t} logP(Y^i \mid X, \theta)$, where $Y^i \in \mathbb{R}^n$ is the vector of the i-th target for each of the n training samples.

2.4 Representing the different products

In bioprocess applications, we want to train a GP on data originating from processes of *different* products, which behave differently. We do not use a detailed description or characterisation of the products but only their names. That is, for each data point we only use the *identity* of the product that it was obtained from. Therefore, we need to add a categorical feature to the GP input, that represents from which of the N_{prods} products the data point was generated. The model then has to discover from the training data alone how (dis)similar the processes of different products behave. Whether this is possible depends on how the product identity is represented as we will see in the following.

2.4.1 Traditional one-hot representation

Traditionally, categorical features are represented by a one-hot vector (similar to dummy variables in statistics) $e^p \in \{0, 1\}^{N_{prods}}$ for $p \in \{1, \dots, N_{prods}\}$ which has a single one at the position corresponding to the product index p. For example the third product would be represented by the vector $e^3 = (0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0)$. The one-hot vector is then appended to the other features. Hence the input feature vector $x = (f \quad e)$ to the GP in Section 2.2 is comprised of process state features $f \in \mathbb{R}^{d'}$ (i.e. current concentration measurements, pH, temperature ...) and the one hot encoding $e \in \{0, 1\}^{N_{prods}}$ of the product.

For a better grasp of the implications we split the sum in (8) to treat the features and the product representation separately:

$$
\begin{aligned}
RBF_\theta(x,x') &= exp\left(-\sum_{i=1}^{d'}\frac{(f_i-f_i')^2}{2\theta_i^2}-\sum_{j=1}^{N_{prods}}\frac{(e_j-e_j')^2}{2\theta_{(d'+j)}^2}\right)\\
&= RBF_\theta(f,f')\cdot exp\left(-\sum_{j=1}^{N_{prods}}\frac{(e_j-e_j')^2}{2\theta_{(d'+j)}^2}\right)\\
&= RBF_\theta(f,f')\cdot RBF_\theta(e,e'),
\end{aligned}
\tag{10}
$$

where e and e' are the one-hot representations of the two products that data points x and x' were generated from.

Hence the kernel has a large value *iff* both the kernel on (f, f') is large *and* the kernel on (e, e') is large. Or in words: x and x' are considered similar *iff* the current concentrations and process conditions are similar *and* the two points come from similar products as measured by $RBF_\theta(e, e')$.

2.4.2 Traditional representation cannot capture pairwise similarities

In general the kernel $RBF_\theta(e, e')$ is a function of the difference vector between inputs $(\mid e_1 - e_1' \mid \quad \dots \quad \mid e_n - e_n' \mid)$ and the vector of length scale parameters θ. However, for one-hot input vectors — which is the case for the second term in (10) — the difference vector is either all 0 (same product) or has exactly two 1-entries in the positions corresponding to the two different products. For the same product we then have a kernel of 1 (perfect similarity) and for different products the kernel is exclusively a function of the two length scale parameters associated with those two different products. More formally, due to the one-hot nature of e, the second term in (10), which measures the similarity between products, can be written as

$$
RBF_\gamma(e,e') = \begin{cases} 1 & \text{if} \quad p = p' \\ exp(-\gamma_p - \gamma_{p'}) & \text{if} \quad p \neq p' \end{cases}, \tag{11}
$$

where we introduced $\gamma_j := \frac{1}{2\theta^2_{d'+j}} \in \mathbb{R}^+$ for notional convenience. Here p and $p' \in \{1, \dots, N_{prods}\}$ denote the ids of the products from which datapoint x and x' respectively came from. Hence, the product similarity measured by (11) only depends on the learned parameters $\gamma_1, \dots, \gamma_{N_{prods}}$ which are determined during the GP optimization procedure (9). That is, the similarity structure between products identified from the training data during the optimization procedure has to be stored in those learned parameters. Therefore, it is prudent to ask, what kind of similarity structures can, in principle, be represented by appropriate values of $\gamma_1, \dots, \gamma_{N_{prods}}$. It turns out that the expressiveness is quite limited. Specifically γ_j can encode how *peculiar* the given product j is compared to all other products, however it cannot encode a more nuanced similarity structure. For example, even a simple clustered structure cannot be represented as can be illustrated with a thought experiment: Assume we have data from $N_{prods} = 100$ different products that form two clusters. Product 1 and 2 in cluster A are very similar to each other but different from products 3 to 100 which form cluster B. Since the products 3 to 100 are very similar to each other, the kernel in (11) should return a large value for any two products from this cluster. This in turn requires that $\gamma_3, \ldots, \gamma_{100}$ are small. Further, the kernel similarity should be small for any product from cluster A with any product from cluster B, which implies that γ_1, γ_2 have to be large. Consequently, the kernel similarity between products 1 and 2 is small. This is not in line with the assumption of the thought experiment. Hence, even for this simple example, the kernel cannot correctly capture the notion of similarity we desire.

2.4.3 The Novel approach - Product embedding

In the previous section we have seen that the parameters of the traditional approach are not sufficiently flexible to allow a useful representation of product similarity. Hence, we seek a novel approach to alleviate this issue. We borrow the concept of (word) *embedding vectors* from natural language understanding where each word is represented by a vector which is used as input to other machine learning models (e.g. a neural network).

Similarly we propose to represent each product $p \in \{1 \ldots N_{prods}\}$ by a learned vector $w^p \in \mathbb{R}^D$, which are organised for convenience as columns in a matrix $W \in \mathbb{R}^{D \times N_{prods}}$. In the input to the GP the sparse and high-dimensional one-hot vector e^p from the previous approach is now replaced with the corresponding (typically) lower-dimensional embedding vector w^p, which has continuous entries (i.e. not 0 or 1). For example all feature vectors in the training set that come from the third product would have their last D entries equal to the third column of the matrix W (i.e. the third embedding vector). Similar to the γ parameters before, the embedding vectors W will be determined during optimization such that they best capture the product similarity observed in the training data. That is, each product is represented by a point in an abstract D-dimensional embedding space.

So how will the RBF kernel behave with this product representation? First note that a *learned* length scale parameter is now redundant for any of the embedding dimensions because it would simply scale (i.e. multiply) the corresponding row of the embedding W which is itself *learned*. As a design choice which does not result in a loss of generality we therefore fix[1] these length scales to 1 and thus the similarity kernel on the product identity (second term in (10)) reads

$$RBF_1(w, w') = exp\left(-\sum_{i=1}^{D} \frac{(w_i - w'_i)^2}{2}\right) = exp\left(-\frac{1}{2}\|w - w'\|^2\right), \tag{12}$$

where $\|\cdot\|$ is the euclidean norm. With this kernel two products are considered similar (high kernel value) *iff* the two associated points in embedding space are close to each other as measured by euclidean distance. Therefore, almost any similarity structure of products can be modelled by choosing appropriate embedding points in a sufficiently high dimensional space. For example, the clustered structure described in the thought experiment in Section 2.4.2 can readily be expressed with $D = 1$. Importantly, clusters are just one example that illustrates the advantage of embedding vectors over the traditional one-hot representation.

Finally we point out that the embedding vectors are invariant to rotations, shifts and mirroring. That is, modifying all embedding vectors by $\widetilde{w}^i := Aw^i + b$ with $A \in \mathbb{R}^{D\times D}$ orthogonal (i.e. $A^T A = \mathbb{I}$) and $b \in \mathbb{R}^D$ does not change kernel values or indeed the predictions of the GP. This further highlights the abstract nature of the embedding space. Only the distances between embedding points matter whereas the exact values of each coordinate axis do not carry any meaning.

2.4.4 Identifying the product embedding vectors

As mentioned before the embedding vectors are determined in the optimization of the hyperparameters using the training data. Specifically, the prior in (6) now depends on W. Hence, in analogy to (9), we choose W together with the length scale vector θ for the process state features according to

$$\theta, W = \underset{\theta', W'}{\arg\max}\, logP(Y|X, \theta', W'). \tag{13}$$

Note that θ here only consists of length scales for the process state features whereas there are no learned length scales for the dimensions corresponding to the product embedding as these are fixed in (12). A convenient way to implement this is by defining a custom kernel with optimisable hyperparameters θ, W as

$$k_{\theta,W}(x, x') = RBF_\theta(f, f') \cdot exp\left(-\frac{1}{2}\|We - We'\|^2\right), \tag{14}$$

where e and e' are one-hot representations of the products which are appended to the training data as in Section 2.4.1. Note that We simply picks the column corresponding to the 1 in e from the matrix

W. This kernel[2] can then be plugged into an existing implementation of the GP hyperparameter tuning and prediction algorithm (e.g. scikit-learn (Pedregosa et al., 2011; Buitinck et al., 2013)).

2.5 Data Generation with an Emulated Bioprocess

Data for N_{prods} different products is generated using the emulator model described in (Narayanan, Behle, et al., 2020). For each product we randomly sampled different values for the parameters of the model equations, such that the process of each product has a characteristic behavior, different to the processes of the other products. To make the process behavior more realistic we allow lactate to be consumed in the processes of some of the products, while it can only be produced in others. We draw experimental designs from a Latin Hyper Cube Design (Mckay, Beckman, & Conover, 2000; Iman, Helton, & Campbell, 1981) with parameter ranges given in Table 1 and simulate experimental runs with the emulator for these designs on the different products, i.e. the experimental design is identical across the products. Concentration measurements of VCD, Glucose, Glutamine, Ammonia, Lactate and Titer are performed once a day in the simulation (these are used to construct the matrix $\mathcal{M}^i$ of measurements in the simulation case) and corrupted with small additive Gaussian noise (standard deviations 0.02; 0.25; 0.02; 0.02; 0.2; 5 respectively) to mimic the analytic measurement error.

2.6 Performance Metric

A test set of $N'_{\mathcal{E}}$ unseen experimental conditions is set apart to assess the prediction quality of the model. The absolute error between predicted concentrations and actual measurements are computed to obtain a $N_{time} \times N_Q \times N'_{\mathcal{E}}$ tensor. To standardise, we divide the absolute errors by the variance in the actual measurements across the $N'_{\mathcal{E}}$ experiments for each time point and quantity. We then take the median across experiments, obtaining a $N_{time} \times N_Q$ matrix. Values well below 1.0 indicate good performance whereas values above 1.0 show that a dummy prediction of the time and quantity dependent mean value would outperform the machine learning model. To provide a rapid understanding of the results we take the average across time to obtain N_Q accuracy values for the measured quantities.

3 Results and Discussion

Wet-lab experiments are cost and time expensive, consequently we are interested in using a model in combination with as few experiments as possible to develop the process for a novel product. In particular, we are interested in improving model accuracy when there is only data from very few experimental runs available. We conjecture that including historic data from other products can increase model accuracy in prediction for the novel product. We investigate this by using a case study with simulated data (Section 2.5): We have 5 *historic* products (**HP1** ... **HP5**) for which we generate data from 16 experimental runs each, where identical experimental conditions are used with each of the 5 products. The behaviour of the historic products is shown in Figure 1 for the first experimental condition of an example data set[3]. Additionally, we have a novel product **NP** for which we have data from only $N_{\mathbf{NP}} \ll 16$ experimental conditions, which are different from the 16 experimental conditions of the historic products. Figure 2 shows the process data from the example data set for the product **NP** with (here $N_{\mathbf{NP}} = 4$) different experimental conditions. More figures visualising the kinetics of the different products can be found in the supplementary material. During each 14-day experimental run measurements are taken after each full day and for example with $N_{\mathbf{NP}} = 4$ runs a total of $4 \cdot 14 = 56$ data points are therefore available from the product **NP**. The trained model is then used according to (2) and (3) to predict the full process development (i.e. six concentration measurements on each day) from the experimental condition. We always measure performance of the trained model by predicting 100 unseen experimental conditions of the novel product **NP**.

3.1 Choosing the Embedding Dimension

The embedding dimension D is chosen based on all data available for training. It has to be large enough such that the appropriate pairwise distances between products can be realised by points in D dimensional space. Hence we choose the value of D at which the log marginal likelihood (13) does not significantly increase any more, which indicates that the optimal pairwise distances can already be realised in this space. Figure 3a shows the likelihood values for the example data set (here $N_{\mathbf{NP}} = 4$). From this we select $D = 3$ because a value of 4 does not further increase the objective.

To show that this way of choosing the embedding dimension is valid, we evaluated the choice using a statistical approach. We independently generate 75 pairs of training ($16 \cdot 5 + 4$ runs) and test (100 runs) sets. Specifically, the experimental conditions for these sets are independently drawn from a Latin Hypercube Design and the processes are then simulated according to Section 2.5. Hence we obtain 75 independent estimates of the test error. Figure 3b shows a box plot comparing the distribution of test error for different choices of D. We observe that the error decreases until $D = 3$

which corroborates that this is actually the choice of D with the lowest expected test error.

3.2 Product Embedding improves Accuracy

The performance of the proposed method is evaluated by predicting test experiments of a novel product **NP**. The models have been trained on data from 16 runs for each of 5 historic products plus data from a varying number $N_{\mathbf{NP}}$ of runs with the novel product. We compare the proposed *product embedding* methodology (Section 2.4.3) with a traditional one hot representation of products (Section 2.4.1). Additionally, we include the baseline ***NP** only* in which a GP is trained with only the $N_{\mathbf{NP}}$ available runs of the novel product. The data for the last baseline therefore has less features, because the representation of the product identity is not needed. We fix a test set of 100 runs on the novel product **NP** and repeat the comparison 150 times with independently generated training sets. This is to ensure statistic validity and avoid reporting results that are merely a consequence of the stochastic variation in one particular training set. Figure 4 shows bar plots of the test error distributions. One can clearly see the benefit of adding data from other historic products when there are only few runs ($N_{\mathbf{NP}} = 4, 6, 8$) of the product of interest available for training, i.e *Product Embedding* and *One Hot* (both using historic data) achieve a lower error than ***NP** only.* Furthermore, the product embedding approach clearly outperforms the one-hot representation of products. For example, with only $N_{\mathbf{NP}} = 4$ experimental runs of the novel product the embedding already achieves similar accuracy as the ***NP** only* baseline with 8 runs or the one hot representation with 6 runs.

As the number of runs of the novel product ($N_{\mathbf{NP}} = 12, 16$) increases, we expect a smaller benefit of using the product embedding with historic data. If enough data for **NP** is available the product's behaviour can directly be modelled sufficiently well without the necessity to extend the data set with related products.

However, it is surprising that for $N_{\mathbf{NP}} = 16$ the ***NP** only* baseline in fact outperforms the embedding algorithm when predicting *titer.* We speculate that this is because the embedding algorithm focuses on modeling the underlying biological system (i.e. the five interrelated quantities *VCD, Glucose, Glutamine, Ammonia, Lactate*) rather than *titer* production, which has no causal influence on the system dynamics. Hence, the embedding would be biased towards grouping products with similar cell characteristics. However, two processes of different products, which exhibit similar cell behaviour (and hence close embedding vectors) could (in our emulator model) still produce *titer* at different levels. Here the similarity information from the embedding would actually be slightly detrimental for performance since it would lead to falsely predicting similar *titer* production as well.

This could explain the better performance of the ***NP** only* algorithm specifically for *titer* prediction.

The performance of *viable cell density* prediction supports this hypothesis (see Figure 5). For this quantity the *Product Embedding* outperforms the baseline for all values of $N_{\mathbf{NP}}$ which indicates that the selected embeddings are indeed more beneficial for reflecting cell characteristics than titer formation. This could possibly be mitigated by having separate embeddings for the task of predicting *titer*. Nonetheless, for both *titer* and *VCD* we see the largest benefit of the product embedding algorithm when there is few data (i.e. $N_{\mathbf{NP}} = 4, 6, 8$).

From the results of the case study the advantage of exploiting historic data with the embedding method for process development becomes evident. It enables us to use an accurate model for making process decisions after just a few experiments. Hence, from a practical workflow perspective, for a novel product one would start with a few runs (calibration experiments - potentially run at conditions at which the model across products shows the greatest divergence), then analysing and modeling the generated data with the embedding method. Subsequently the embedding model should be exploited to design the next set of experiments such that the information density around the process conditions of interest is maximized. Once enough maximally informative data is generated for the novel product it can be analyzed with either the embedding method or the NP only method, dependent on the purpose of the analysis. Likewise, the exploitation of the model for design, optimization or control would dictate which modeling approach is more suited for the application at hand, i.e. balancing the local accuracy versus the general description of the behaviour across products. In any case, the embedding approach will help to reduce the requirement for wet-lab experiments with the novel product, placing the experiments where it matters, thereby accelerating process development while decreasing costs.

3.3 Interpretable Embedding Vectors

We have argued in Section 2.4.3 that products are similar if the associated embedding points have small euclidean distance. The embedding vectors are initialised uniformly at random in the cube $[0.1 \quad 3]^3$ and during training of the GP the *L-BFGS-B* (Zhu, Byrd, Lu, & Nocedal, 1997) algorithm is used with this initialisation to find the (possibly local) optimum of the objective (13). That is, each product is associated with a point (embedding vector) in the abstract embedding space such that the distances between points capture the similarities between products.

Here, we are particular interested in the distances between the novel product **NP** and each of the historic products **HP1**, . . . **HP5**. Figure 6 shows the distribution of distances in the embedding space before (left) and after (right) training. Clearly the distances of the trained embeddings are very different from the random initialisation. In particular the small inter quartile ranges indicate that the embedding captures similarity between products in a reliable way that is invariant to the training set used. That is, across the 150 training sets there is only slight variation in the distances between the learned embedding vectors.

Specifically, we see that products **HP1** and in particular **HP2** are most similar to **NP**. This is not surprising as these three products all have lactate consuming behaviour whereas products **HP3**, **HP4** and **HP5** do not.

The similarity structure observed in Figure 6, can also be found by looking at the product embedding vectors learned from one data set (Figure 7). This embedding was obtained using the example training set (same as Figure 3a) and hence represents a practical application of the algorithm with a real data set, i.e. as when using one data set during process development. It demonstrates that the lactate consuming products **HP1**, and **NP** form a visually inferred group around **HP2**, whereas the products **HP3** and **HP5** form a looser group around **HP4**. We also clearly see that **HP2** is very similar to the novel **NP**. In the development of a novel process such insights can help to understand the process behavior of the novel product and its relationship to historic, well studied processes. This itself is a valuable use case of the product embedding algorithm in addition to the improved predictive accuracy.

The qualitative structure exemplified by Figure 7 is statistically relevant and found in most of the 150 repetitions. To confirm this, we include in the supplementary material plots similar to Figure 6 that show the distribution of distances from each of the 5 historic products.

3.4 Limitations and Future Work

The setting of this simulation study was deliberately kept simple to allow a clean and concise introduction to the concept of product embedding vectors in Gaussian Process regression. As such there are a number of limitations that require future research.

First, the level of measurement noise was fixed for all our simulations. With increasing noise level we would naturally expect a decrease in predictive performance for all presented regression models. However we do not have reason to believe that the product embedding method is particularly susceptible to measurement noise. Nonetheless, this should be explored in depth in future research.

Second, we always use the same experimental conditions for all five historic products, because in the industry experimental designs are many times identical across products as platform processes are used. This is just a simplifying assumption and not necessary for the proposed method. Indeed the process runs for the novel product were always obtained with a different set of experimental conditions. Therefore we stipulate that similar results as the presented ones should be obtained also when different designs are used across products, provided that all designs cover a comparable space of experimental conditions. To which degree nuanced design choices influence the performance of the method should be investigated thoroughly in the future.

Lastly, our case study does not include the comparison with other regression algorithms. The goal of this paper is to show how historic data from different products can be used more effectively with the proposed embedding approach *specifically* in GP regression. The adaptation of the product embedding method to the potentially more optimal neural network learning for bioprocess modelling is straightforward.

4 Conclusions

We propose an embedding approach for multiple product spanning probabilistic process data modeling and evaluate it using an upstream mammalian (cell-culture) bioprocess simulation case study. In particular, we use a hybrid regression model to predict the evolution of the bioprocesses.

We demonstrate that prediction performance of the hybrid model can be improved by including data from other, historic products, using the proposed embedding approach. This creates a model capable of transferring knowledge from data of processes which behave similar and thereby reduces the data required with the novel process to train an accurate model. We show that visualising the embedding space can offer valuable insights for experts about the complex relation between products and processes. More importantly, during prediction it enables the algorithm to give higher importance to information from those data points in the training set that were obtained with similar behaving products. For this reason the embedding algorithm outperforms the traditional one-hot representation, as was shown with the simulation study.

The findings are highly useful in process development where one aims to find process conditions that produce a desired outcome (e.g. high *titer* production). The model can already be trained to high accuracy after obtaining data from only few experiments with the novel process by leveraging available historical data from different processes. For instance, it has been shown that with only four experiments on the novel product, one could reach a similar model precision to the case where eight to twelve experiments were used without any knowledge transfer.

The trained model can then be used in a variety of ways to determine the next process conditions that should be evaluated in a wet-lab experiment. In the simplest greedy scheme one could choose the process conditions for which the trained model predicts the highest expected *titer*. More elaborate methods could utilise the uncertainty of the Gaussian Process regression model to balance exploitation and exploration. The optimal procedure for this has been a subject of research in computer science and statistics (Auer, 2003; Auer, Cesa-Bianchi, & Fischer, 2002; Srinivas, Krause, Kakade, & Seeger, 2009) but an algorithm tailored to bioprocess development requires future considerations. Thus concluding, we believe that the proposed algorithm will be able to significantly speed up bioprocess development, or will allow the gain in predictability to significantly improve the quality of such processes.

Notes

[1] Assume we did not fix the length scales for the embedding dimensions to 1. Then the embedding vectors w and a length scale θ_i for each dimension of w_i are determined simultaneously during the hyperparameter optimization. We would not expect these length scales to equal 1. However a simple change of variable yields an equivalent situation where indeed $\theta_i = 1$. Specifically note that the length scales always enter the kernel in the form $\frac{w_i}{\theta_i}$. Hence setting $w_i \leftarrow \frac{w_i}{\theta_i}$ (for all embedding vectors w) and $\theta_i \leftarrow 1$ does not change kernel values or the GP predictions. This corresponds to scaling the i-th row of the matrix W which contains all embedding vectors as columns. Now as both w_i and θ_i are learned simultaneously during hyperparameter optimization this implies the following: The optimum of the objective with regard to w_i (with $\theta_i = 1$ fixed) has the same value as the optimum of the objective with regard to both w_i and θ_i. Hence we do not need the extra parameters wherefore we fix those length scales to 1.

[2] An implementation can be found at `https://github.com/rauwuckl/EntityEmbedding4GP.git`

[3] The example dataset is provided at `https://github.com/rauwuckl/BioprocessExampleData.git`

Data Availability

- The core part of the code is available at `https://github.com/rauwuckl/EntityEmbedding4GP`

- The example data set used in figures 1, 2, 3 and 7 is available at `https://github.com/rauwuckl/BioprocessExampleData`

References

Abt, V., Barz, T., Cruz, N., Herwig, C., Kroll, P., Möller, J., … Schenkendorf, R. (2018, 12). *Model-based tools for optimal experiments in bioprocess engineering* (Vol. 22). Elsevier Ltd. doi: 10.1016/j.coche.2018.11.007

Anane, E., Garcia, A. C., Haby, B., Hans, S., Krausch, N., Krewinkel, M., … Cruz Bournazou, M. N. (2019, 11). A model-based framework for parallel scale-down fed-batch cultivations in mini-bioreactors for accelerated phenotyping. *Biotechnology and Bioengineering*, *116*(11), 2906–2918. doi: 10.1002/bit.27116

Auer, P. (2003, March). Using confidence bounds for exploitation-exploration trade-offs. *J. Mach. Learn. Res.*, *3*(null), 397–422. doi: 10.5555/944919.944941

Auer, P., Cesa-Bianchi, N., & Fischer, P. (2002, 5). Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, *47*(2-3), 235–256. doi: 10.1023/A:1013689704352

Brendel, M., & Marquardt, W. (2008, 7). Experimental design for the identification of hybrid reaction models from transient data. *Chemical Engineering Journal*, *141*(1-3), 264–277. doi: 10.1016/j.cej.2007.12.027

Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., … Varoquaux, G. (2013). API design for machine learning software: experiences from the scikit-learn project. In *Ecml pkdd workshop: Languages for data mining and machine learning* (pp. 108–122).

Bushnell, G. G., Rao, S. S., Hartfield, R. M., Zhang, Y., Oakes, R. S., Jeruss, J. S., & Shea, L. D. (2020, 1). Microporous scaffolds loaded with immunomodulatory lentivirus to study the contribution of immune cell populations to tumor cell recruitment in vivo. *Biotechnology and Bioengineering*, *117*(1), 210–222. doi: 10.1002/bit.27179

Ferreira, A. R., Dias, J. M., Von Stosch, M., Clemente, J., Cunha, A. E., & Oliveira, R. (2014, 9). Fast development of Pichia pastoris GS115 Mut+ cultures employing batch-to-batch control and hybrid semi-parametric modeling. *Bioprocess and Biosystems Engineering*, *37*(4), 629–639. doi: 10.1007/s00449-013-1029-9

Gori, M. (2018). Chapter 1 - the big picture. In M. Gori (Ed.), *Machine learning* (p. 2-58). Morgan Kaufmann. doi: 10.1016/B978-0-08-100659-7.00001-4

Hans, S., Haby, B., Krausch, N., Barz, T., Neubauer, P., & Cruz-Bournazou, M. N. (2020, 9). Automated conditional screening of multiple Escherichia coli strains in parallel adaptive fed-batch cultivations. *Bioengineering*, *7*(4), 1–18. doi: 10.3390/bioengineering7040145

Iman, R. L., Helton, J. C., & Campbell, J. E. (1981). An Approach to Sensitivity Analysis of Computer Models: Part I-Introduction, Input Variable Selection and Preliminary Variable Assessment. *Journal of Quality Technology*, *13*(3), 174–183. doi: 10.1080/00224065.1981.11978748

Karlberg, M., von Stosch, M., & Glassey, J. (2018, 8). *Exploiting mAb structure characteristics for a directed QbD implementation in early process development* (Vol. 38) (No. 6). Taylor and Francis Ltd. doi: 10.1080/07388551.2017.1421899

Kuchemüller, K. B., Pörtner, R., & Möller, J. (2020). Digital Twins and Their Role in Model-Assisted Design of Experiments. In (pp. 1–33). Springer, Berlin, Heidelberg. doi: 10.1007/10_2020_136

Lee, S. L., O'Connor, T. F., Yang, X., Cruz, C. N., Chatterjee, S., Madurawe, R. D., ... Woodcock, J. (2015, 9). *Modernizing Pharmaceutical Manufacturing: from Batch to Continuous Production* (Vol. 10) (No. 3). Springer New York LLC. doi: 10.1007/s12247-015-9215-8

Li, F., Vijayasankaran, N., Shen, A., Kiss, R., & Amanullah, A. (2010, 9). *Cell culture processes for monoclonal antibody production* (Vol. 2) (No. 5). Taylor & Francis. doi: 10.4161/mabs.2.5.12720

Li, X., Brock, G. N., Rouchka, E. C., Cooper, N. G., Wu, D., OToole, T. E., ... Rai, S. N. (2017, 5). A comparison of per sample global scaling and per gene normalization methods for differential expression analysis of RNA-seq data. *PLoS ONE*, *12*(5), e0176185. doi: 10.1371/journal.pone.0176185

Liu, X., Li, N., Liu, S., Wang, J., Zhang, N., Zheng, X., ... Cheng, L. (2019, 11). Normalization Methods for the Analysis of Unbalanced Transcriptome Data: A Review. *Frontiers in Bioengineering and Biotechnology*, *7*, 358. doi: 10.3389/fbioe.2019.00358

Mckay, M. D., Beckman, R. J., & Conover, W. J. (2000). A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output From a Computer Code. *Technometrics*, *42*(1), 55–61. doi: 10.1080/00401706.2000.10485979

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013). Distributed representations ofwords and phrases and their compositionality. In *Advances in neural information processing systems* (pp. 1–9).

Mikolov, T., Yih, W.-t., & Zweig, G. (2013, June). Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 conference of the north American chapter of the association for computational linguistics: Human language technologies* (pp. 746–751). Atlanta, Georgia: Association for Computational Linguistics. Retrieved from `https://www.aclweb.org/anthology/N13-1090`

Narayanan, H., Behle, L., Luna, M. F., Sokolov, M., Guillén-Gosálbez, G., Morbidelli, M., & Butté,

A. (2020, 9). Hybrid-EKF: Hybrid model coupled with extended Kalman filter for real-time monitoring and control of mammalian cell culture. *Biotechnology and Bioengineering*, *117*(9), 2703–2714. doi: 10.1002/bit.27437

Narayanan, H., Luna, M. F., Stosch, M., Cruz Bournazou, M. N., Polotti, G., Morbidelli, M., . . . Sokolov, M. (2020, 1). Bioprocessing in the Digital Age: The Role of Process Models. *Biotechnology Journal*, *15*(1), 1900172. doi: 10.1002/biot.201900172

Narayanan, H., Sokolov, M., Morbidelli, M., & Butté, A. (2019, 10). A new generation of predictive models: The added value of hybrid models for manufacturing processes of therapeutic proteins. *Biotechnology and Bioengineering*, *116*(10), 2540–2549. doi: 10.1002/bit.27097

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., . . . Duchesnay, E. (2011, November). Scikit-Learn: Machine Learning in Python. *J. Mach. Learn. Res.*, *12*(null), 2825–2830. doi: 10.5555/1953048.2078195

Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global vectors for word representation. In *Emnlp 2014 - 2014 conference on empirical methods in natural language processing, proceedings of the conference* (pp. 1532–1543). doi: 10.3115/v1/d14-1162

Politis, S. N., Colombo, P., Colombo, G., & Rekkas, D. M. (2017, 6). *Design of experiments (DoE) in pharmaceutical development* (Vol. 43) (No. 6). Taylor and Francis Ltd. doi: 10.1080/03639045.2017.1291672

Portela, R. M. C., Varsakelis, C., Richelle, A., Giannelos, N., Pence, J., Dessoy, S., & von Stosch, M. (2020). When Is an In Silico Representation a Digital Twin? A Biopharmaceutical Industry Approach to the Digital Twin Concept. In *Advances in biochemical engineering/biotechnology* (pp. 1–21). Berlin, Heidelberg: Springer. doi: 10.1007/10_2020_138

Rameez, S., Mostafa, S. S., Miller, C., & Shukla, A. A. (2014, 5). High-throughput miniaturized bioreactors for cell culture process development: Reproducibility, scalability, and control. *Biotechnology Progress*, *30*(3), 718–727. doi: 10.1002/btpr.1874

Rasmussen, C. E., & Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. Cambridge, MA, USA: MIT Press.

Richelle, A., Lee, B. W., Portela, R. M. C., Raley, J., & Stosch, M. (2020, 8). Analysis of Transformed Upstream Bioprocess Data Provides Insights into Biological System Variation. *Biotechnology Journal*, 2000113. doi: 10.1002/biot.202000113

Schweidtmann, A. M., Rittig, J. G., König, A., Grohe, M., Mitsos, A., & Dahmen, M. (2020, 9). Graph Neural Networks for Prediction of Fuel Ignition Quality. *Energy & Fuels*, *34*(9), 11395–11407. doi: 10.1021/acs.energyfuels.0c01533

Shrirao, A. B., Kung, F. H., Omelchenko, A., Schloss, R. S., Boustany, N. N., Zahn, J. D., ... Firestein, B. L. (2018, 4). Microfluidic platforms for the study of neuronal injury in vitro. *Biotechnology and Bioengineering*, *115*(4), 815–830. doi: 10.1002/bit.26519

Srinivas, N., Krause, A., Kakade, S. M., & Seeger, M. (2009, 12). Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design. *IEEE Transactions on Information Theory*, *58*(5), 3250–3265. doi: 10.1109/TIT.2011.2182033

Stosch, M. v. (2018, 9). Hybrid Models and Experimental Design. In *Hybrid modeling in process industries* (pp. 37–61). CRC Press. doi: 10.1201/9781351184373-3

Teixeira, A., Alves, C., Alves, P., Carrondo, M., & Oliveira, R. (2006, 10). Dynamic optimisation of a recombinant BHK-21 culture based on elementary flux analysis and hybrid parametric/nonparametric modeling. *Microbial Cell Factories*, *5*(Suppl 1), S25. doi: 10.1186/1475-2859-5-S1-S25

Velugula-Yellela, S. R., Kohnhorst, C., Powers, D. N., Trunfio, N., Faustino, A., Angart, P., ... Agarabi, C. (2018, 9). Use of high-throughput automated microbioreactor system for production of model igG1 in CHO cells. *Journal of Visualized Experiments*, *2018*(139), 58231. doi: 10.3791/58231

von Stosch, M., Davy, S., Francois, K., Galvanauskas, V., Hamelink, J.-M., Luebbert, A., ... Glassey, J. (2014, 6). Hybrid modeling for quality by design and PAT-benefits and challenges of applications in biopharmaceutical industry. *Biotechnology Journal*, *9*(6), 719–726. doi: 10.1002/biot.201300385

von Stosch, M., Oliveira, R., Peres, J., & Feyo de Azevedo, S. (2014, jan). Hybrid semi-parametric modeling in process systems engineering: Past, present and future. *Computers & Chemical Engineering*, *60*, 86–101. doi: 10.1016/j.compchemeng.2013.08.008

Yu, L. X., & Kopcha, M. (2017, 8). The future of pharmaceutical quality and the path to get there. *International Journal of Pharmaceutics*, *528*(1-2), 354–359. doi: 10.1016/j.ijpharm.2017.06.039

Zhu, C., Byrd, R. H., Lu, P., & Nocedal, J. (1997, 12). Algorithm 778: L-BFGS-B: Fortran Subroutines for Large-Scale Bound-Constrained Optimization. *ACM Transactions on Mathematical Software*, *23*(4), 550–560. doi: 10.1145/279232.279236

List of Figures

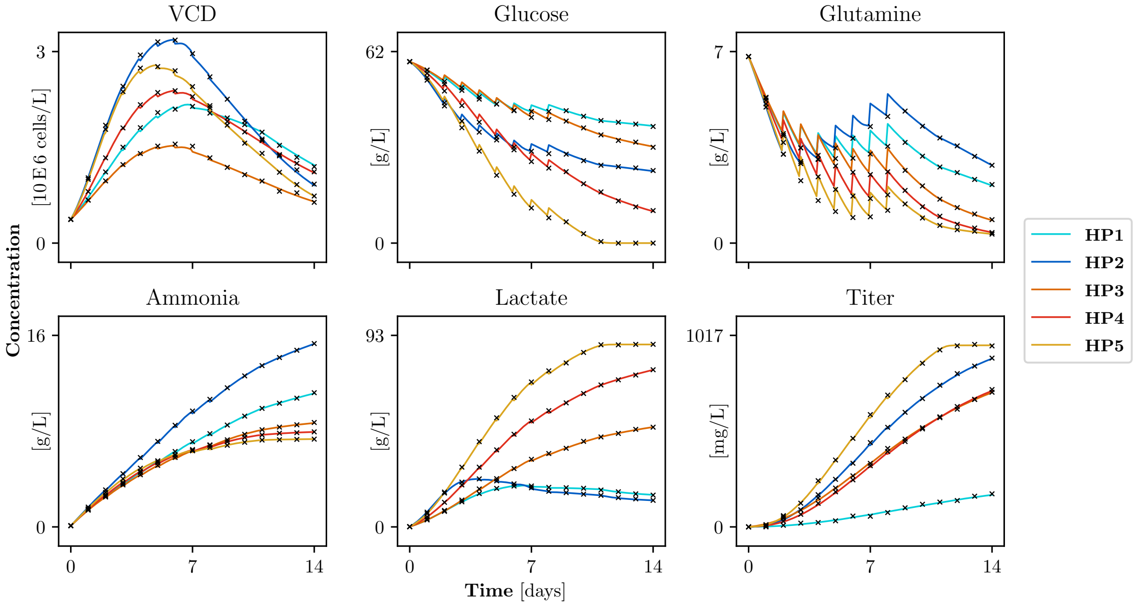

Figure 1: Simulated process data of the 5 historic products for the first experimental condition. The markers × represent measurements which are taken after each full day. Only this data is available for model training. The full process evolution is shown with continuous lines to illustrate the bolus feed events (here on days 2-8). Clearly the processes of the products behave different in response to the same experimental condition. Note in particular the Lactate consuming behaviour of products **HP1** and **HP2**.

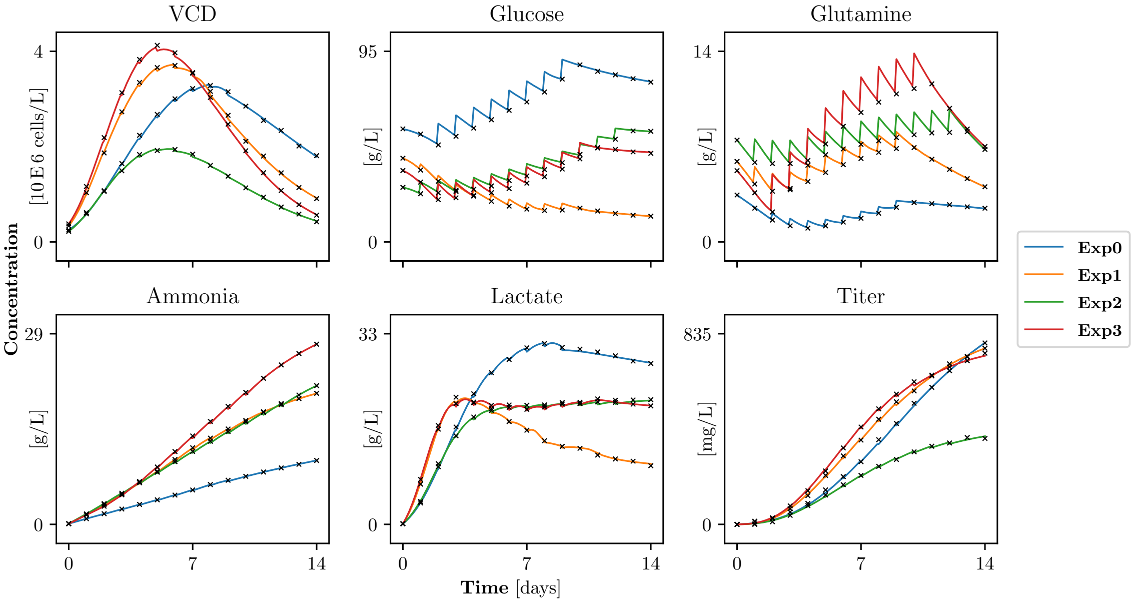

Figure 2: Simulated process data of the novel product **NP** for 4 different experimental conditions. The variability between the runs illustrates the diversity of experimental conditions.

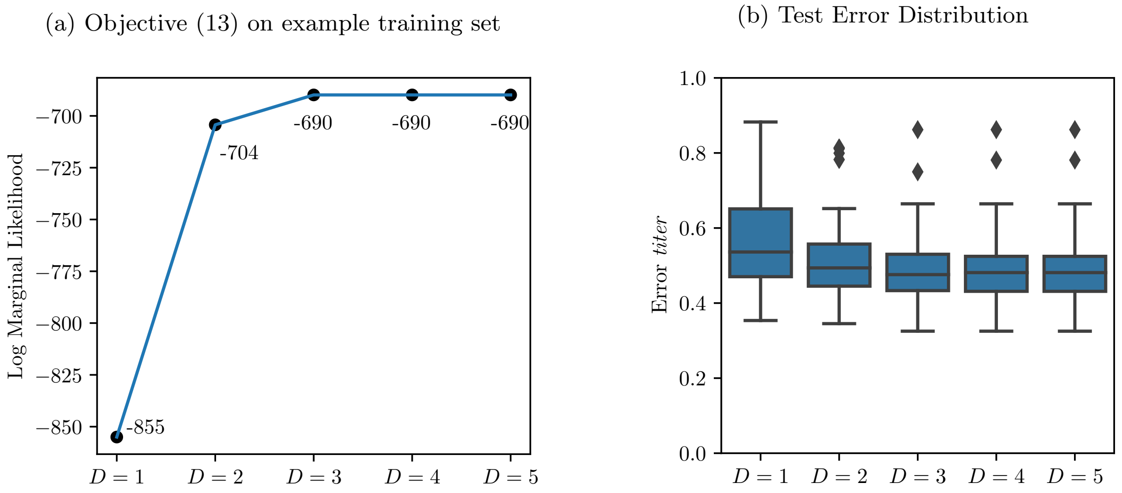

Figure 3: **(a)** In practice the highest log marginal likelihood (13) of the training set can be used to select $D = 3$. **(b)** We validated this choice by showing the test error distribution obtained with 75 different training and test sets. It confirms that the test error does not improve after $D = 3$.

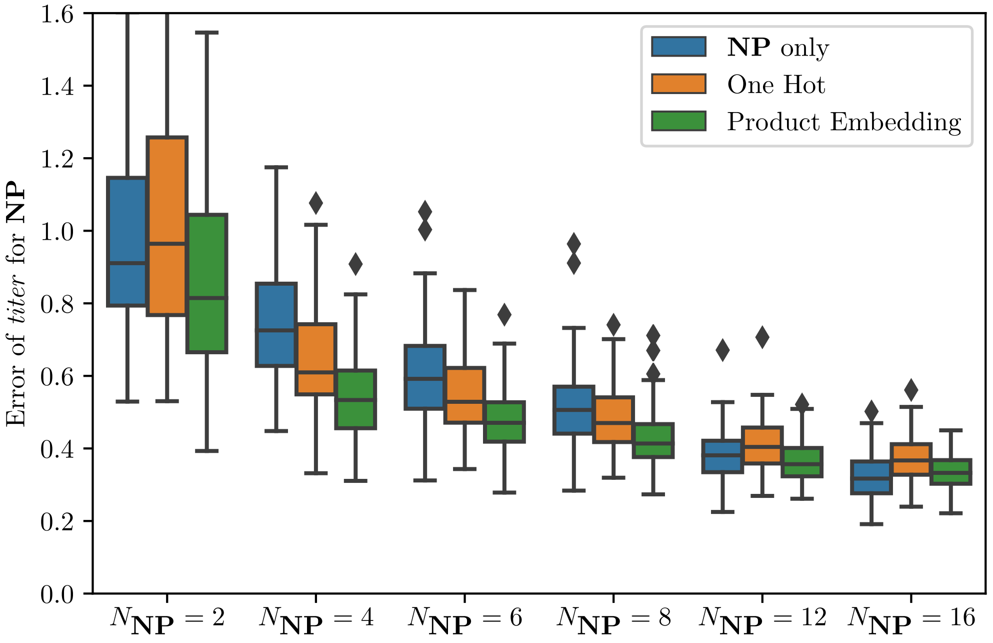

Figure 4: The test error distribution for the product **NP** across 150 random draws of the training set. We vary $N_{\mathbf{NP}}$, the number of runs on the product **NP** available for training, and try three different algorithms, i.e.: *Product Embedding* is the methodology proposed in Section 2.4.3; *One Hot* is the traditional one-hot representation of the product (Section 2.4.1) and for ***NP** only* we only use the data of the $N_{\mathbf{NP}}$ runs available for the novel product.

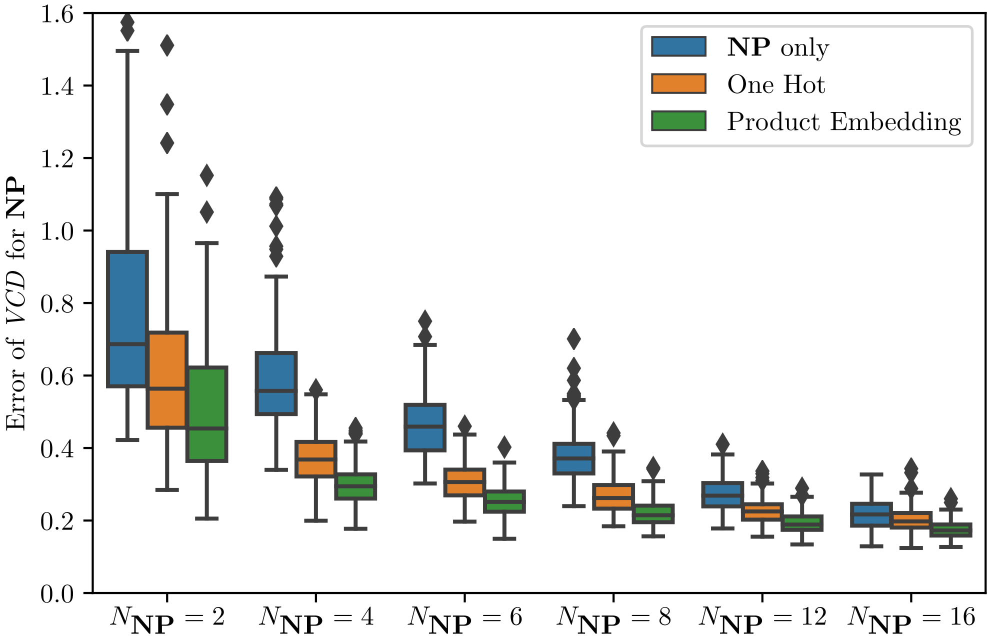

Figure 5: Test error distribution over 150 draws of the training set for predicting *Viable Cell Density* concentrations.

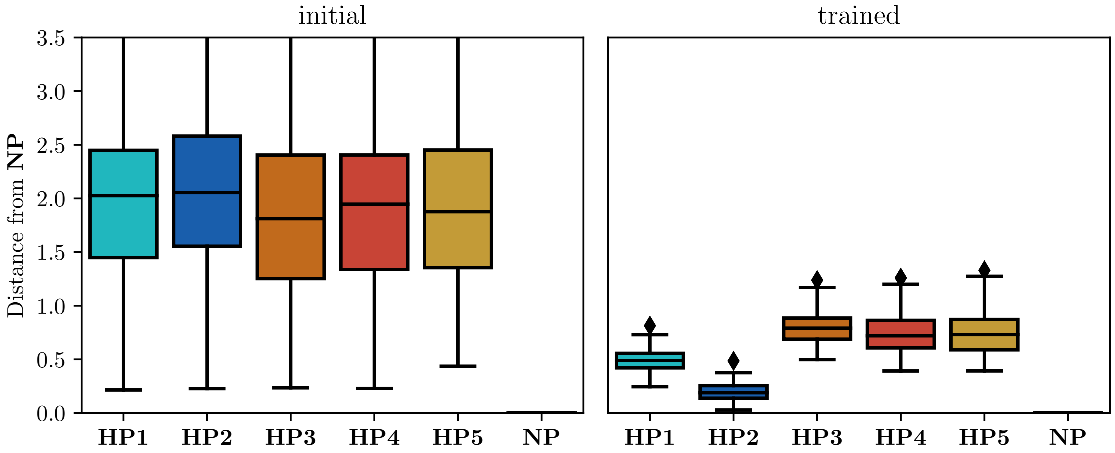

Figure 6: Distribution of distances from the embedding point of the product **NP** to the embedding points of **HP1**. . . **HP5** across 150 repetitions. The embeddings are determined during the optimization procedure (Section 2.4.4) such that the distances best capture the similarities between products observed in the training data.

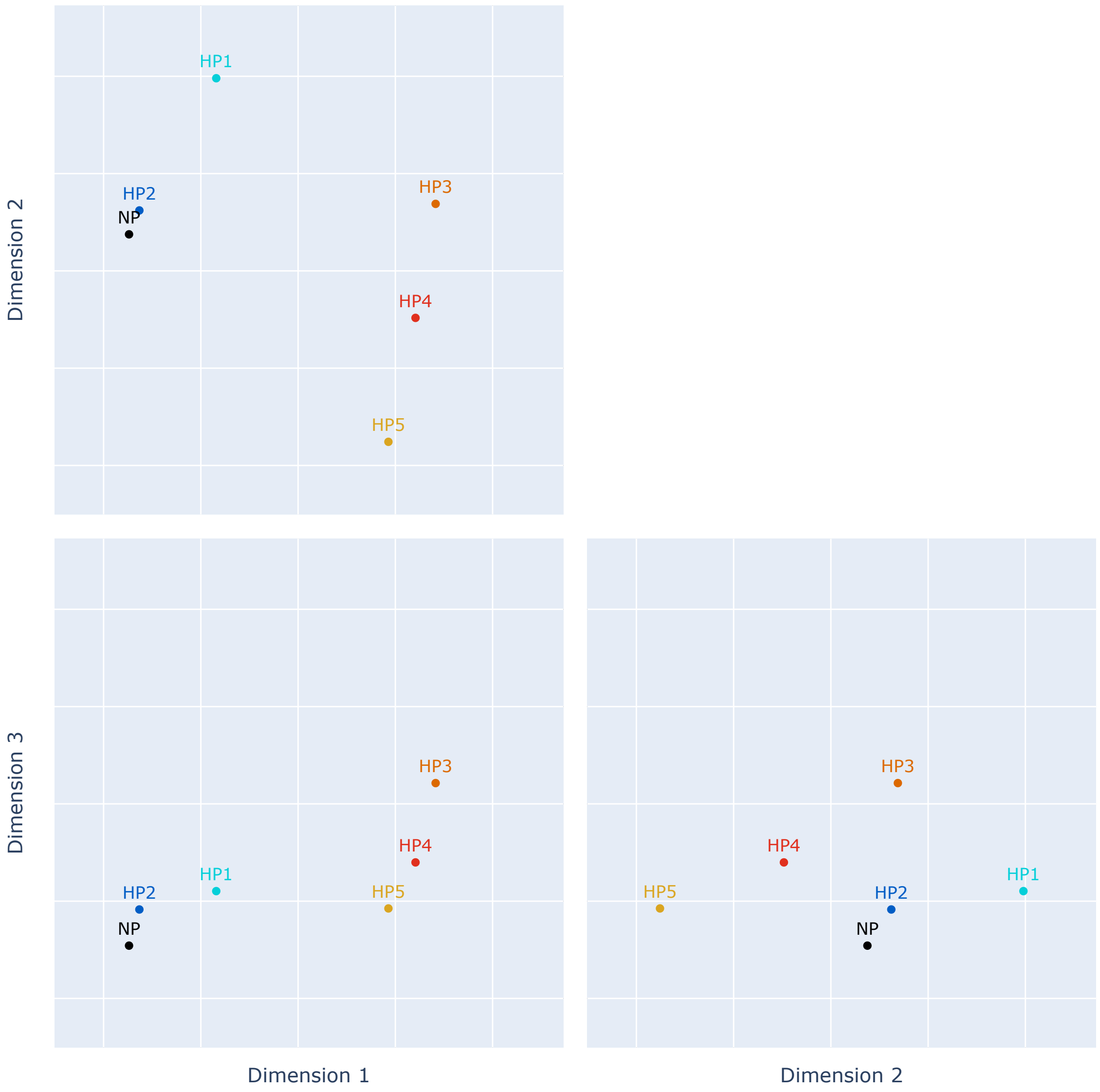

Figure 7: One example of a product embedding. During the optimization procedure each product is associated with a point (embedding vector) in an abstract embedding space such that the distances between points best capture the similarities between products as they are observed in the training data. The dimensions of the embedding space do not carry any specific physical meaning rather the information is encoded in the distances between points. See Section 2.4 for more details.

List of Tables

	Min	**Max**	**Unit**
Initial VCD	0.2	0.4	1e6 cells/L
Initial Glucose Concentration	20	60	g/L
Initial Glutamine Concentration	3	8	g/L
Initial Ammonia Concentration*	0.1		g/L
Initial Lactate Concentration*	0.1		g/L
First Day of Feed	1	3	days
Last Day of Feed	8	13	days
Volume of Feed*	0.005		L
Feed Mass Glucose	0.5	3	g
Feed Mass Glutamine	0.01	0.75	g
Temperature First Half	36	37.5	°C
Time of Temperature change	6	14	days
Temperature Second Half	35	37	°C
pH First Half	6.9	7.1	
Time of pH change	6	14	days
pH Second Half	6.7	7	
Stirring Rate	150	200	RPM
Experiment Duration*	14		days
Measurement Volume*	0.005		L
Initial Bioreactor Volume*	0.25		L

Table 1: Parameter ranges for the experimental designs. Parameters marked with * are fixed for all experimental conditions.