# Get a Human-In-The-Loop: Feature Engineering via Interactive Visualizations

Dimitra Gkorou[1], Maialen Larrañaga[2], Alexander Ypma[1], Faegheh Hasibi[3], Robert Jan van Wijk[1]

[1] ASML, Veldhoven, the Netherlands
{dimitra.gkorou,alexander.ypma,robert-jan.van.wijk}@asml.com
[2] Tessella, Toulouse, France
maialen.mlz@gmail.com
[3] Radboud University of Nijmegen, the Netherlands
f.hasibi@cs.ru.nl

**Abstract.** In manufacturing, data sets tend to be high-dimensional, with a low number of labels and, features show spurious correlations with respect to a target *key performance indicator*. As a consequence, costly manual feature engineering by domain experts is required prior to prediction. To improve this process, we propose an interactive feature engineering scheme based on dimensionality reduction. Low-dimensional embeddings of selected features are visualized and guide the domain experts towards effective feature engineering. We show that by engineering features we obtain higher predictive capabilities and we improve the interpretability of the model.

## 1 Introduction and Background

Many applications of predictive Machine Learning (ML) require significant Feature Engineering (FE) when having small datasets. Particularly in Integrated Circuit (IC) manufacturing, which is the application of this work, the absence of large amounts of labeled data, the requirements of interpretability and the already mature domain knowledge make FE crucial for predictive tasks. Recently, deep learning has shown potentiality in automatically generating useful features. However, the data requirements for obtaining good accuracy with deep learning i.e., about 5000 labeled instances for decent performance and about 10 million labeled instances for outstanding performance [1, Chap. 1], are not realistic for IC manufacturing. Also, most of our usecases require interpretability because ML predictions are expected to contribute to decisions on fab processes with high financial impact and so, highly complex models are not applicable. Finally in IC manufacturing, FE typically requires knowledge on the physics of a process which cannot be easily obtained by statistical techniques. As a result, FE is a costly process which is performed by domain experts manually.

We propose an interactive FE scheme, with a human expert in-the-loop, based on state-of-the-art dimensionality reduction. We implemented an initial approach of it as a web application in ASML, the leading manufacturer of lithography machines and major player in the semiconductor industry. Our scheme is an iterative process. In each iteration an expert observes an embedding of selected features and acquires some information based on the cluster structure

© 2020 for this paper by its authors. Use permitted under CC BY 4.0.

CEUR-WS.org/Vol-2660/ialatecml_shortpaper4.pdf

present in it. For example, they understand which features are responsible for the clusters in the embedding or they understand what context the clusters belong to. After having observed the embedding, they then provide a set of rules (e.g., a clustering). This information is used to engineer a new feature. Expert input is obtained through visualizations. The human-defined clustering encodes (1) their prior knowledge on the underlying predictive task and, (2) the knowledge acquired through unexpected or surprising structure observed in the embedding. The previously observed patterns are factored out from the embedding. A new iteration begins with the expert observing the new embedding for clustering that can used for a new feature. The interaction ends when at a given iteration the embedding shows no more relevant clustering.

**Related Work** Our work has been motivated by the works in `Tiler` [3], and `SIDE` [7]. These pioneering works construct *informative* visualizations that are tailored for each particular user, based on their prior knowledge. However, these works consider linear dimensionality reduction (DR) which is not suitable for the complexity of our data sets. When using linear DR, we often see no cluster structure. [8] and [4] propose non linear DR for informative visualizations: conditional Variational Autoencoders and conditional t-SNE respectively. These methods can be used in our interactive scheme in order to construct embeddings that guide domain experts towards feature engineering.

**Use case** IC manufacturing is a complex process where various machines and processing tools are used such as coating, exposure or etching tools. ICs are being fabricated on a thin silicon plate, called *wafer*, which is processed in several *layers*. Sensors monitor each step of this process. The raw measurements of these sensors are the *features* used to predict *Key Performance Indicators* (KPIs). As an example of KPI, the work in [2] aims to predict the precision in *nanometers* of printing IC designs on a wafer, called *overlay*, using sensor measurements and context information. Overlay is measured over several positions on a wafer after each process layer. The features have a direct physical relationship with the KPI. The tools associated with the sensors are the *context* of the measurements. Unlike features, the context variables such as tool names, machine settings, time stamps, do not have a direct physical relationship with the KPIs. Nevertheless, context variables are necessary in order to explain the raw sensor measurements [5]. Predicting KPIs only based on the raw measurements gives low accuracy models. Typically, sensor measurements are noisy, with redundancies, and have offsets depending on their context namely, associated tools used in a particular step. Moreover, the tools are not always matched with respect to a common reference. For these reasons, enhancing these models with features properly engineered by domain experts is crucial. To evaluate our interactive scheme, we consider the prediction of a KPI in [2] which is a typical use case of our domain.Detailed description of the use case and the features can be found in [2].

## 2 Feature Engineering with Human in the Loop

In this section, we describe the proposed methodology for feature, how the interactive scheme is implemented and the improvements we obtain by engineering the features during the interaction.

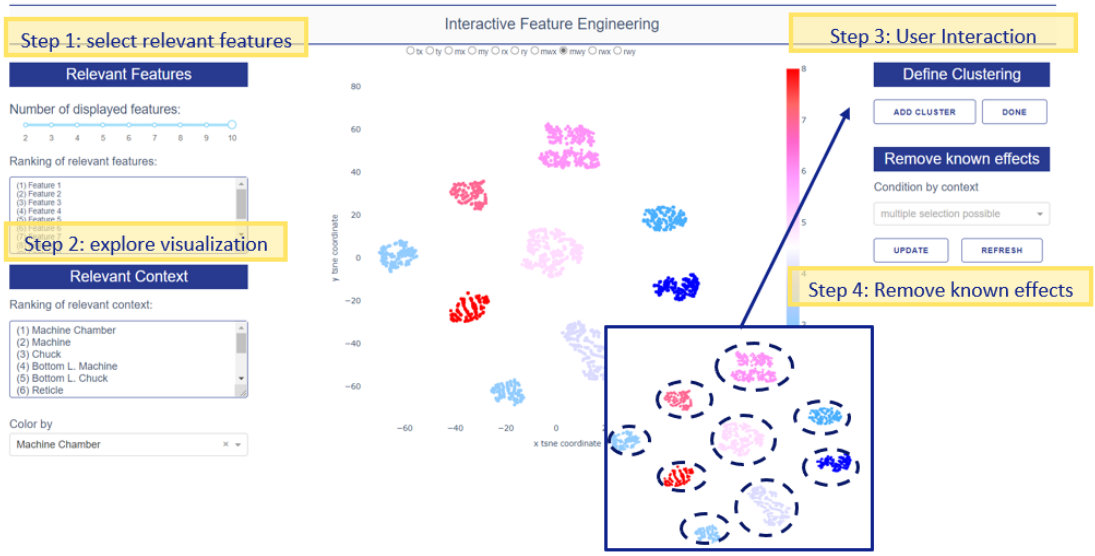


Fig. 1: Software design and steps to follow in the interactive scheme.

### 2.1 Methodology and SW tool

We developed a web application tool in `DASHPlotly` in python 3.7. It has four main steps as shown in Fig. 1. Below we describe it in detail.

**Step 1:** [*Select relevant features and visualize data*] Raw features together with context information is given as input to the software in a tabular format. Then an algorithm ranks the most relevant features with respect to the target KPI. Feature selection is out of the scope of this work because any feature selection could be used without affecting the FE interactions. For the feature selection, we use Bayesian Regression [6]. The user can select the number of ranked relevant features that will be used for the visualization, in Fig. 1 10 are picked. Based on those 10 features data is visualized in 2D using well-known dimensionality reduction methods. In Fig. 1, we used t-SNE as we have a small dataset. In a setting with large datasets, dimensionality reduction techniques such as UMAP [9] and Variational Autoencoders [8] can be used.

**Step 2:** [*Explore visualization*] On the left panel, the user can choose the context with which the scatter plot is colored. It can be the machine where the wafer has been exposed(which is selected in Fig. 1), the Reticle that has been used in the exposure, etc. The expert sees what context explains the clustering or structure in the data. To facilitate this, the context variables are ranked according to their Mutual Information with the clustering on the embedding as defined by Hierarchical Density-Based Spatial Clustering (HDBSCAN). So the user can start exploring the embedding using the context that correlates the most to its structure. As an example of the knowledge that the expert can acquire in this phase, we can see the *context effects* in Fig. 1 which can be explained by the machine chamber settings.

**Step 3:** [*User interaction for Feature Engineering*] The clustering observed in the previous step can be added as cluster constraints by the user. In this way, the user engineers a feature by computing the cluster-offset per machine setting,

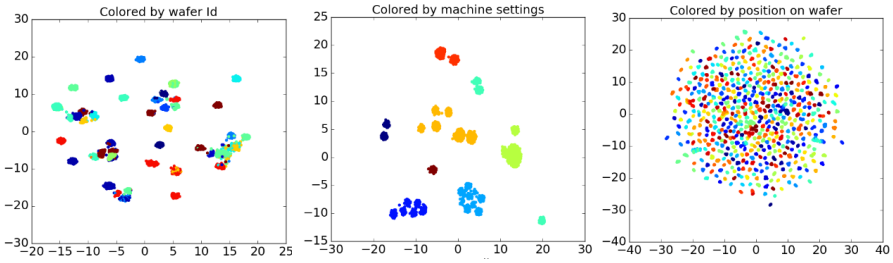


Fig. 2: Each embedding refers to an iteration of the interactive FE scheme.

namely the average of the target KPI per cluster. In fact, the new feature is a form of *target encoding* the machine context variable.

**Step 4:** [*Remove known effects*] The user continues the exploration in order to discover less dominant context-effects. To achieve this, we consider the previously acquired knowledge as prior information and we factor it out from the embedding using conditional t-SNE [4]. In the panel in Fig.1, the user is able to select the context-effect to be removed. Multiple selections are possible. The interaction starts again with a new visualization.

The interaction stops when the user cannot make sense out of the structure on the data anymore. In that step, we assume that every effect that could be *easily* understood from the data is known by the human expert and a feature has been engineered for it.

### 2.2 Application to the Overlay Prediction Use Case

We now present the iterations of human interactions with the proposed FE scheme for the KPI (overlay) prediction use case described in Section 1. First, we describe what a human expert learned by observing the embedding and what kind of features were engineered. Then, we evaluate the impact of the engineered features on overlay prediction.

We have a data set that consists of $\sim$ 2000 wafers. Overlay is measured in 60 points on each wafers and so, in total, our data has 120,000 datapoints. Overlay is a continuous value and thus, we have a regression problem. After feature selection process, from an initial data set of $\sim$ 350 features, we have 30 features. To avoid overfitting, we used $\sim$ 10% of data for FE and the rest of it for training models. In Fig. 2 we see each iteration of the FE scheme.

**Iteration 1:** The data in the first plot of Fig. 2 is colored by wafer Id, which explains most of the clustering structure. This means that the overlay measurements on a wafer (independently of the layer they have been measured) look similar. In our data set, the measurements on a wafer are always taken with respect to the same reference layer. This means that if there is a distortion in the reference layer it will propagate through stack to all the layers above. Overlay experts can quickly identify this behavior. They now need to make sure that the initially loaded data contains the overlay measurements of at least one previous layer per wafer or the reference layer. In this case, two features can be engineered; the estimated cluster average and the overlay measurements of a previous layer.

**Iteration 2:** The structure observe in the 2*nd* embedding in Fig. 2 is obvious for an expert who can quickly relate it to overlay. It belongs to 8 different machine settings. Here the engineered feature is simply the estimated averages per setting.

Table 1: Prediction performance ($r^2$) for three iterations of the FE scheme.

| | Bayesian Ridge | Random Forest |
|---|---|---|
| Baseline | 0.0107 | 0.4537 |
| 1st iteration | 0.3057 | 0.4636 |
| 2nd iteration | 0.5863 | 0.601 |
| 3rd iteration | 0.589 | 0.6027 |

**Iteration 3:** The last iteration shows a clustering that is colored per position on wafer. It is known that measurements of errors on the edge of a wafer are larger than those on the interior rings of the wafer. This structure again is easy for an expert to relate to overlay errors. The engineered feature is just the expected error on each position of the measurement.

In order to evaluate the accuracy of the model we implement a linear and a non-linear ML algorithm; Bayesian Ridge Regressor and Random Forest from scikit-learn. In Table 1, we evaluate two ML algorithms at the different phases of the interaction. The reported results derive from a 3-fold cross validation.

The first row, i.e., *baseline*, refers to the accuracy obtained by using the raw input data after feature selection. We see that the linear regressor was not able to capture the contributors to overlay because typically the signals are related to the overlay in a non-linear fashion. Random forest is able to capture quite some of the effect with the raw input data. In the next three rows, from *1st iteration* to *3rd iteration*, in each step new features have been added. We see that, the more features we engineer the better $r^2$ values we obtain. Another key message is that, once we have engineered all features, the results by a linear machine and a non-linear learning machine become comparable. In this example, the accuracy that we have obtained is not really high ($r^2 \sim 0.6$). Overlay prediction is a challenging task, and a result of 0.6 after a few FE steps is quite good.

**Why do we need interpretable features?** In this example, experts could easily identify the structures in each iteration. They were able to iteratively obtain all three *effects* contributing to the target KPI; distortions on reference layer, scanner settings and measurement position. Typically, context effects are dominant and can be easily identified by experts in data. However, some data sets might have more complex clustering structures that domain experts cannot relate with context variables. In IC manufacturing, a field heavily relying on the physics of the process, it is preferred to have an interpretable and less predictive feature than a more predictive but not well understood one. Experts usually discard predictive features that cannot explain as spurious correlations. In the proposed FE scheme, the interpretability of features by the experts is a requirement. At least some of the clusters of the embedding have to be explained by the context variables in order to engineer a feature.

**Why do we need a human expert?** One could argue that if the clustering in the visualization is obvious, a clustering algorithm such as DBSCAN could be run to do FE automatically. However, as we have seen in Iteration 1, the engineered features where not simple cluster average estimations, we also engineered another feature based on the knowledge an expert has on the domain. Also, clustering is an ill-posed problem in the sense that different clustering algorithms or different initializations of the same algorithm might give different

results on the same data. Having an expert in the loop makes sure that not only the hidden structure in the data will be properly captured by engineered features, but also the domain knowledge will be used. The model becomes more interpretable, since a few human defined features are used as input data.

## 3 Conclusions and Open Questions

Our proposed FE scheme facilitates experts to engineer features in a structured way using their domain knowledge. The proposed scheme lends itself very naturally for semiconductor applications, but may be just as applicable in situations with high complexity, small sample sizes and existence of relevant (but maybe implicit) domain knowledge, e.g. medicine, general industry settings, etc. Nevertheless, we still face several challenges in implementing the proposed FE scheme:

First, we would like to *transfer* the learned features from one domain to another. Similar context effects are present in many predictive machine learning settings over different domains within semiconductor industry. The reason is that we typically want to predict KPIs from sensor measurements associated with some context. Our FE scheme requires costly time from domain experts. Transferring the learned features across domains, instead of requesting similar input from expert in different domains, will improve its efficiency.

Second, domain experts are often unsure of their feedback and they might also be biased. Making our scheme robust to biases and conflicting inputs is necessary for its successful adoption.

## References

1. I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.
2. F. Hasibi, L. van Dijk, M. Larrañaga, A. Pastol, A. Lam, and R. van Haren. Towards fab cycle time reduction by machine learning-based overlay metrology. In *34th European Mask and Lithography Conference*, pages 129 – 137. SPIE, 2018.
3. A. Henelius, E. Oikarinen, and K. Puolamäki. Tiler: Software for human-guided data exploration. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 672–676. Springer, 2018.
4. B. Kang, D. García García, J. Lijffijt, R. Santos-Rodríguez, and T. De Bie. Conditional t-sne: Complementary t-sne embeddings through factoring out prior information, 2019.
5. A. Lam, A. Ypma, M. Gatefait, D. Deckers, A. Koopman, R. van Haren, and J Beltman. Pattern recognition and data mining techniques to identify factors in wafer processing and control determining overlay error. *Proc. SPIE*, 9424, 2015.
6. M Larranaga, D Gkorou, T Guzella, A Ypma, F Hasibi, and RJ van Wijk. Towards Interactive Feature Selection with Human-in-the-loop. In *Workshop on Interactive Adaptive Learning*, 2018.
7. J. Lijffijt, B. Kang, K. Puolamäki, and T. De Bie. Side: a web app for interactive visual data exploration with subjective feedback. In *ACM SIGKDD Workshop on Interactive Data Exploration and Analytics (IDEA)*, 2016.
8. A. Marot, A. Rosin, L. Crochepierre, B. Donnot, P. Pinson, and L. Boudjeloud-Assala. Interpreting atypical conditions in systems with deep conditional autoencoders: the case of electrical consumption. In *ECML PKDD*, 2019.
9. L. McInnes, J. Healy, and J. Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.