Towards Explanatory Model Monitoring

Alexander Koebler[1,2*] **Thomas Decker**[1,3*] **Michael Lebacher**[1] **Ingo Thon**[1]
Volker Tresp[3,4] **Florian Buettner**[1,2,5,6†]
[1]Siemens AG [2]Goethe University Frankfurt [3]LMU Munich
[4]Munich Center for Machine Learning (MCML)
[5]German Cancer Research Center (DKFZ) [6]German Cancer Consortium (DKTK)
{alexander.koebler,thomas.decker,michael.lebacher,
ingo.thon,buettner.florian}@siemens.com
volker.tresp@lmu.de

Abstract

Monitoring machine learning systems and efficiently recovering their reliability after performance degradation are two of the most critical issues in real-world applications. However, current monitoring strategies lack the capability to provide actionable insights answering the question of why the performance of a particular model really degraded. To address this, we propose Explanatory Performance Estimation (XPE) as a novel method that facilitates more informed model monitoring and maintenance by attributing an estimated performance change to interpretable input features. We demonstrate the superiority of our approach compared to natural baselines on different datasets. We also discuss how the generated results lead to valuable insights that can reveal potential root causes for model deterioration and guide toward actionable countermeasures.

1 Introduction

Deploying Machine Learning (ML) models successfully in practice is a challenging endeavor as it requires models to cope well with complex and dynamic real-world environments [36, 30]. Consequently, monitoring and maintaining ML-models has been established as a central pillar of the modern ML-Life cycle [38, 25] and commercial ML frameworks [29]. A crucial assumption to assure the validity of a model is that the data distribution during training matches the real-time distribution during deployment. However, this assumption might be violated in real-world applications for various reasons hard to identify given the black-box nature of the observed system. For instance, data integrity issues such as hardware deterioration or modifications in the collection and processing pipeline could cause a mismatch as well as intrinsic changes in the data generation process due to novel real-world circumstances. Since any potential discrepancy might compromise the reliability of predictions, continuous assessment of the model and its input data is required. For this purpose, many different approaches have been proposed [34] that can conceptually be divided into two main categories. Performance monitoring methods [11, 22] enable systematic tracking of the model performance over time and provide an early indication of significant model deterioration. However, such approaches typically require access to ground truth labels at inference time, which is usually infeasible or quite expensive. In contrast, unsupervised data drift detection [35, 12] quantifies to which extent the input data characteristics have changed to identify distribution shifts irrespective of the actual performance. In practice, many different root causes could underlie an observed distribution shift with individual implications. While expensive retraining might be unavoidable in case of an intrinsic change in the relationship between input

[*] Equal contribution
[†] Work done for Siemens AG

XAI in Action: Past, Present, and Future Applications @ NeurIPS 2023.

data and output labels, it would be ineffective for mitigating problems arising from hardware failure such as a defective sensor. Accordingly, feedback from practitioners [9, 37] suggests that maintaining reliable ML-systems requires high ML expertise, as current monitoring approaches do not provide truly actionable insights that guide users to efficient remediation when degradations occur. In this work, we introduce Explanatory Performance Estimation (XPE) that systematically addresses the desired needs of actionable model monitoring in practice. In particular, we propose a framework that anticipates the performance change caused by an observed distribution shift and guides users toward potential root causes and actions.

Problem Setting We consider the common situation where a machine learning model $f : \mathcal{X} \to \mathcal{Y}$ has been trained to perform a prediction task in a supervised fashion based on labeled training data $\{(x_s^i, y_s^i)\}_{i=1}^{n_s}$. Further, we assume the training data originates from a source distribution denoted as $P_s(X, Y)$. At some point during deployment, we suppose that the underlying data distribution changes and further equals to the target distribution $P_t(X, Y)$ with $P_t(X, Y) \neq P_s(X, Y)$. As common in practice, we suppose that during deployment we only have access to unlabeled data instances $\{x_t^i\}_{i=1}^{n_t}$ originating from the marginal target distribution $P_t(X)$. The overall goal of XPE is twofold. First, it should provide a reasonable estimate of the model's performance under $P_t(X, Y)$ despite missing target labels. Second, it should be able to identify through which specific input features the distribution shift affects the model.

Related Work Although the connection between model monitoring and feature attributions seems intuitive, only a limited number of works have focused on this intersection. Amazon's SageMaker Model Monitor [29] or Google's Vertex AI Model Monitoring [40] offer monitoring of feature attributions and interpret changing importance scores as an indicator for potential performance degradation. In [27] the authors demonstrate on synthetic tabular examples that monitoring attribution results can be superior compared to monitoring input data characteristics. However, it remains unclear under which circumstances this approach can reliably signal an actual performance decrease. Another related approach is simply to combine drift detection with attribution methods and expect a performance change if an important feature shifts, as mentioned in [16]. But feature attributions on drifted data might produce unreliable results, and models can also be robust to certain shifts even on important features, implying no performance loss. In [4], Shapley Values are leveraged to identify potential reasons for a distribution shift based on causal graphs, and [47] applies this idea in the context of model monitoring. While theoretically appealing, these approaches heavily rely on complete knowledge about the causal mechanisms of the true data-generating process which is infeasible to attain in practice.

2 Explanatory Performance Estimation

In this section, we formalize our approach, which aims to reveal through which specific features an observed distribution shift affects a model during deployment.

Aligning distributions via optimal transport A natural way to gain a better understanding of how a distribution shift precisely impacts the predictions of a machine learning model is to systematically compare its individual predictions before and after the shift happened. For this purpose, suppose that the experienced distribution shift can be expressed by a functional transformation T, such that $P_t(X) = P_s(T^{-1}(X))$. In this case, the immediate effect on a single model prediction $f(x)$ that is purely induced by the distribution shift can be analyzed by comparing the corresponding predictions $f(x)$ and $f(T(x))$. Modern deep neural networks have demonstrated impressive capabilities to parameterize functional transformations that perform complex and realistic distribution shifts [32, 31, 26]. However, they typically require a lot of data to be trained and might even rely on labeled source/target pairs. During deployment, there is usually only a limited number of discrete samples from the source and target domain available. Therefore, we consider a more feasible way to estimate the relationships between P_s and P_t in practice based on Optimal Transport (OT) [42]. In general, OT refers to the mathematical problem of identifying the most cost-efficient way to transform one probability measure into another and has already been applied to various related tasks such as semantic alignment of different data structures [18, 21, 15] or domain adaptation [6, 7, 5]. To demonstrate its adequacy for model monitoring suppose we have n_s samples randomly drawn from the source domain

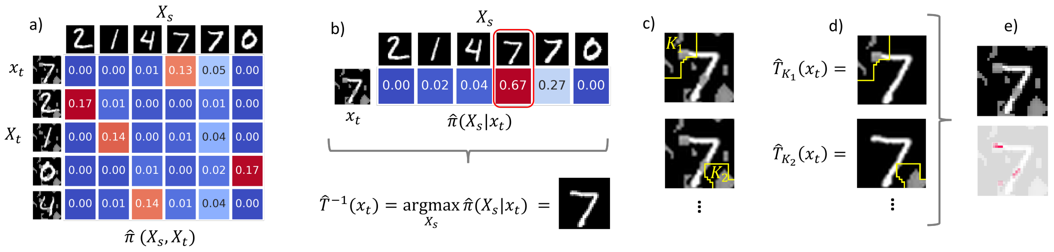


Figure 1: Overview of our proposed framework for Explanatory Performance Estimation (XPE): **a)** Based on optimal transport, an optimal coupling is estimated to sample-wise align empirical source and target distributions. **b)** For a given target sample x_t the conditional coupling $\hat{\pi}(X_s|x_t)$ indicates the most likely version of x_t in the source domain denoted by $\hat{T}^{-1}(x_t)$. **c)** Given pre and post-shift version $\hat{T}^{-1}(x_t)$ and x_t one can restrict shifts to individual input feature subsets K_i and **d)** simulate partial feature shifts $\hat{T}_{K_i}(x_t)$ by replacing x_t with $\hat{T}^{-1}(x_t)$ outside the considered regions. **e)** Finally, all simulated partial shifts can be aggregated to quantify how individual feature shifts have contributed to the anticipated model loss based on Shapley values.

$\Omega_s = \{x_s^i\}_{i=1}^{n_s}$ and n_t from the target domain $\Omega_t = \{x_t^i\}_{i=1}^{n_t}$. Let δ_x be the Dirac delta function, describing a valid probability distribution concentrated at the point x, then the empirical source and target distributions are given by:

$$\hat{p}_s = \sum_{x_s \in \Omega_s} \frac{1}{n_s} \delta_{x_s} \qquad\qquad \hat{p}_t = \sum_{x_t \in \Omega_t} \frac{1}{n_t} \delta_{x_t}$$

Given a non-negative cost function $c : \Omega_s \times \Omega_t \to \mathbb{R}^+$, the relationship between $\hat{p}_s$ and $\hat{p}_t$ can be expressed via a probabilistic coupling π representing any joint distributions over $(\Omega_s \times \Omega_t)$ with marginals equal to $\hat{p}_s$ and $\hat{p}_t$. This leads to the discrete Kantorovich formulation of OT which estimates a cost-efficient alignment of source and target samples:

$$\hat{\pi} = \arg\min_{\pi \in \Pi} \sum_{x_s \in \Omega_s} \sum_{x_t \in \Omega_t} c(x_s, x_t)\pi(x_s, x_t) \quad \text{with} \quad \Pi = \{\pi \in \mathbb{R}^{n_s \times n_t} \mid \pi \mathbf{1}_{n_t} = \hat{p}_s, \pi^T \mathbf{1}_{n_s} = \hat{p}_t\}$$

Hence, searching for an optimal coupling results in a linear program that can be solved directly using appropriate solvers [33]. However, it's worth noting that there are computationally more efficient strategies available, for instance using entropic regularization [8]. Intuitively, $\hat{\pi}$ provides a probabilistic estimate of how samples of the source domain are likely to look in the target domain and vice versa (see Fig. 1a) if the observed shift is cost-minimizing with respect to c. This equips us with an appealing tool to comprehend the precise nature of the shift and can further be utilized to reveal how an observed shift affected a model. In this case, understanding the impact of a distribution shift for a single prediction $f(x_t)$ could be achieved by comparing it with all predictions corresponding to the potential source version of x_t as implied by the conditional coupling $\hat{\pi}(X_s|x_t)$ (Fig.1b). Moreover, it is straightforward to transform a probabilistic alignment into a deterministic one by matching each source sample with its most related target sample. This results in a transform $\hat{T}(x_s) = \arg\max_{x_t \in \Omega_t} \hat{\pi}(x_t|x_s)$ and equivalently $\hat{T}^{-1}(x_t) = \arg\max_{x_s \in \Omega_s} \hat{\pi}(x_s|x_t)$ mapping each x_t onto its most likely source version. Note that the resulting coupling depends on the chosen cost function c, for which we consider the squared Euclidean distance as it is the most popular choice in practice.

Shapley values for feature shift importance Shapley values have been introduced as a fair way to distribute the total outcome of a coalition game to individual players $D = \{1, \ldots d\}$. In this context, a game can be specified via a value function $v(K) : 2^D \to \mathbb{R}$ that quantifies the value that each possible subset or coalition of players $K \subseteq D$ would achieve if only they would contribute. Given a value function, the Shapley value of each player $i \in D$ results as a weighted average of its marginal contributions over all possible coalitions and orders:

$$\phi_i = \sum_{K \subseteq D \setminus \{i\}} \frac{1}{\binom{d-1}{|K|} d} v(K \cup \{i\}) - v(K)$$

For the purpose of feature attribution given a model f, individual features resemble players, and $v(K)$ is defined as a hypothetical model prediction where only features in K would be present. Different computational methods have been proposed to enable such a value function by simulating model predictions under feature absence [24, 39]. In order to better understand how an observed input feature shift influenced a model prediction we propose to consider a novel coalition game where $v(K)$ expresses the model prediction under the assumption that only features in K did experience the shift (Fig.1c). As introduced above, optimal transport allows us to identify potential pre- and post-shift versions of data instances related to the empirical source and target distributions. The corresponding results can directly be utilized to perform the required partial distribution shifts of a given target sample x_t (Fig.1d). If the shift is due to a transformation T, then the desired value function is given by:

$$v_T(K) = f(T_K(x_t)) \text{ with } T_K(x_t) = \left(x_t^K, T^{-1}(x_t)^{K^c}\right)$$

where K^c is the complement of the index set K and x^K denotes all entries of x with index in K. When computing Shapley values with respect to this value function, ϕ_i can be interpreted as a measure of how the empirical shift of feature i contributed to the shift-related prediction change (Fig.1e). Moreover, carefully comparing v_T with the corresponding formulation used for feature attributions reveals a close relationship. While $v(K)$ is designed to resemble partial feature absence for the purpose of feature attribution, our proposed version estimates feature shift importance by simulating partial feature shifts. It can even be considered as a specific variant of standard Shapley values for feature importance [39, 23], where the baseline is set according to the results of optimal transport to capture the effect of a distribution shift explicitly.

Explaining an anticipated performance change When the data distribution changes, it is critical to reevaluate whether the model still performs well under the new circumstances. Reliably computing the performance of a model would require access to corresponding ground truth labels. Such information is typically unavailable during deployment and usually requires cumbersome manual efforts. However, empirically aligning labeled source samples Ω_s and unlabeled target samples Ω_t via transformation T also equips us with a reasonable way to anticipate the performance by supposing that all linked instances exhibit the same label. More precisely, one can obtain for any $x_t \in \Omega_t$ a label estimate $\hat{y}_t$ by allocating the known label of the linked source sample $T^{-1}(x_t) \in \Omega_s$. This strategy has also already been successfully leveraged to enable unsupervised domain adaptation [7]. By combining transport-based label estimation with feature shift importance, we are now able to specify our Explanatory Performance Estimation (XPE) approach. Given a loss function $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ we define a new value function $v^{\mathsf{XPE}}(K) = \mathcal{L}\left(f\left(T_K(x_t)\right), \hat{y}_t\right)$ which expresses directly the anticipated performance change under partial feature shifts. The corresponding Shapley values ϕ^{XPE} finally indicate through which specific features an observed distribution shift impacts the anticipated performance providing valuable information regarding potential root causes of model degradation.

3 Evaluation and Experiments

Natural baselines To assess the capabilities of XPE we first formalize natural baselines that are connected to existing model monitoring practices. Remember that XPE aims to evaluate through which specific features an observed distribution shift impacts the model performance. A first baseline for this purpose is simply to check whether predictions in the target domain tend to rely on other features compared to the source domain. Let $\phi(x)$ be the outcome of standard Shapley values explaining the prediction $f(x)$ for an instance x. Given an estimated transportation map $\hat{T}$, we can simply compare the explanations of two matched samples individually and define a **local attribution difference** (LAD):

$$\phi^{\mathsf{LAD}}(x_t, \hat{T}) = |\phi(x_t) - \phi\big(\hat{T}^{-1}(x_t)\big)|$$

If this difference is large for a specific feature then the distribution change in this feature might be particularly harmful. Note that this method also relates to the existing practice of monitoring changes in feature attributions [29, 40, 27] proposed to detect model degradation during deployment. A second baseline that does not require an empirical alignment of source and target samples is first to perform unsupervised drift detection and consider only shifted features which are also important for the model as intermediaries of the shift. For this purpose, we leverage a two-sided Kolmogorov-Smirnov (KS) test to assess whether the distribution of each feature within the source samples is significantly different from the corresponding one of the target samples. Let $M^{KS} \in \{0,1\}^d$ be a binary mask indicating which of the d input features have drifted according to the KS-test. Then, the **Attribution** × **Shift** (AxS) baseline is given by:

$$\phi^{\text{AxS}}(x_t, M^{KS}) = \phi(x_t) \odot M^{KS}$$

Defining suitable metrics Quantitatively evaluating any kind of model explanation is challenging, but a desirable property is faithfulness [3]. It generally tries to asses if perturbing features with high attribution scores also cause coherent prediction changes and a variety of different related metrics have been proposed [1, 45, 3]. To evaluate feature shift attributions, we reformulate the faithfulness criterion in the following way: When features with high shift attributions are shifted back, we expect the model performance to recover equivalently. Suppose access to the true pre-shift version $T^{-1}(x_t)$ of a target sample x_t as well as to the ground truth source and target labels y_s and y_t. Then, we can define **Shift-Faithfulness** (S-Faith) of a feature shift attribution ϕ^{Shift} as the correlation between the actual performance change under partial feature shift and the sum of allocated shift importance:

$$\text{S-Faith}(\phi^{\text{Shift}}, f, x_t, T, y_t, y_s) = \underset{K \in \binom{d}{|K|}}{\text{corr}} \left(\sum_{i \in K} \phi_i^{\text{Shift}}, \mathcal{L}\big(f(x_t), y_t\big) - \mathcal{L}\big(T_{K^c}(x_t), y_s\big) \right)$$

Here we adapted the metric based on the notation from [3], where the correlation is computed using different feature subsets K with fixed size $|K|$. Another popular metric to measure the quality of feature attributions is RemOve And Retrain (ROAR) [14] assessing if the performance actually decreases when important features are removed and models retrained. Consequently, we propose an adapted metric coined **remove, retrain, and shift (ROAR-S)** which evaluates whether the performance decrease caused by a shift diminished if features with high shift importance are removed and the model retrained. More precisely, we define the ROAR-S score as the proportion of shift-induced performance decrease that remains when for each instance the top 5% of input features highlighted by ϕ^{Shift} are removed and the model subsequently retrained. If this score is small, the distribution change no longer affects the performance and the shift importance is reliable. More details about this metric are deferred to the supplementary material. To quantify the practical information content of explanations, we consider the **Complexity** (Cpx) metric [3], which is defined as the Shannon entropy H of the normalized attribution values: $\text{Cpx}(\phi^{\text{shift}}) = H(|\phi^{\text{shift}}| / \sum_i |\phi_i^{\text{shift}}|)$. This expresses the uncertainty of shift attribution results across all input features and lower values indicate that the method is able to communicate the potential reason for model degradation more concisely. This makes the results more comprehensible to humans and helps to identify concrete countermeasures [2].

3.1 Experiments

The goal of our experiments is to rigorously analyze feature shift attributions and their capabilities to intuitively explain the true model behavior under deployment-related distribution shifts. Therefore, we designed an appropriate evaluation setup to investigate how data quality issues affect deep learning-based image classification models. For our implementation, we relied on several popular open-source tools [24, 10, 41, 13].

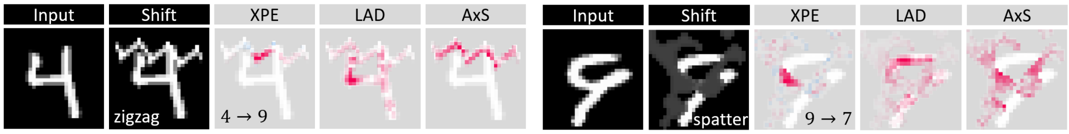
Quantitative evaluation We consider a variety of popular lightweight image datasets [19, 43, 44] and simulated several distribution shifts mimicking potential camera-related hardware degradation or physical changes in the environment [28]. This setup ensures complete knowledge about the true pre- and post-shift pairs, which is crucial to reliably evaluating the quality of shift attribution methods via Shift-Faithfulness. For each dataset, we fitted a LeNet model [20] and evaluated Shift-Faithfulness based on 500 test samples and the cross-entropy loss as a performance measure. The average results are reported in Table 1 and indicate that XPE almost consistently outperforms all baselines. The baselines are often not correlated at all with the true performance change induced by the shift of highlighted features. The corresponding results for Complexity imply that the explanations of XPE also tend to be the most concise given a sufficient degree of faithfulness. For cases where other methods provide significantly less complex results,

		brightness		contrast		dotted		fog		gaussian		impulse		spatter		zigzag		AVG
		S-Faith↑	Cpx↓	S-Faith↑	Cpx↓	S-Faith↑	Cpx↓	S-Faith↑	Cpx↓	S-Faith↑	Cpx↓	S-Faith↑	Cpx↓	S-Faith↑	Cpx↓	S-Faith↑	Cpx↓	ROAR-S↓
MNIST	XPE	**0.45**	**4.03**	**0.53**	5.91	**0.82**	**2.55**	**0.52**	5.88	**0.71**	**3.71**	**0.76**	**2.53**	**0.69**	**4.32**	**0.78**	**3.38**	**0.37**
	LAD	0.01	5.62	0.32	**5.16**	0.22	4.70	0.15	**5.69**	0.11	5.25	0.17	4.69	0.21	5.05	0.13	4.82	2.12
	AxS	-0.04	5.42	0.24	5.81	0.31	3.12	0.11	5.92	0.07	4.92	-[1]	-[1]	0.26	4.91	0.21	3.82	1.87
FashionM	XPE	**0.58**	**5.86**	**0.58**	**5.91**	**0.79**	**2.82**	**0.72**	**5.97**	**0.73**	**5.24**	**0.79**	**3.19**	**0.72**	**4.85**	**0.76**	**3.51**	**0.20**
	LAD	0.00	6.12	0.06	6.05	0.04	5.63	-0.00	6.11	0.04	5.96	0.02	5.62	0.03	5.82	0.05	5.67	0.77
	AxS	-0.01	5.99	0.01	5.93	0.07	3.24	0.01	6.04	0.03	5.72	-[1]	-[1]	0.09	5.04	0.07	3.88	0.78
OrganaM	XPE	**0.32**	5.54	**0.25**	5.48	**0.39**	5.16	**0.23**	5.54	**0.33**	5.52	**0.41**	5.15	**0.34**	5.29	**0.40**	5.23	**0.48**
	LAD	0.05	5.63	0.09	5.33	0.06	5.54	0.04	5.68	0.05	5.58	0.04	5.55	0.06	5.62	0.04	5.57	0.60
	AxS	-0.00	**4.88**	0.10	**4.25**	0.06	**4.66**	0.02	**4.98**	0.00	**4.82**	0.02	**4.77**	0.01	**4.95**	0.04	**4.73**	0.56
PneumM	XPE	**0.52**	5.05	**0.28**	4.72	**0.72**	**4.39**	**0.28**	4.84	**0.54**	4.87	**0.69**	**4.02**	**0.59**	4.55	**0.59**	4.86	**0.96**
	LAD	-0.01	5.37	0.01	5.00	0.13	5.62	-0.00	4.89	0.03	5.29	0.09	5.43	0.03	5.43	0.00	5.62	2.00
	AxS	-0.01	**4.86**	0.07	**3.90**	0.13	4.63	-0.00	**3.05**	0.03	**4.57**	0.09	4.29	0.05	**4.52**	-0.01	**4.77**	5.05

Table 1: Average S-Faith, Cpx, and ROAR-S results of shift attributions methods for a LeNet on different image datasets and corruptions. A higher S-Faith value indicates that features highlighted by ϕ^{Shift} are stronger correlated with the true performance change caused by the shift in these features. A lower Complexity value corresponds to more concise explanations and a lower ROAR-S score signals that removing features based on ϕ^{Shift} effectively mitigates the shift-induced performance change. [1] The KS-Test did not identify any shift, so AxS is all zeros.

they typically have almost no correlation with the actual performance decrease. Moreover, the label transport accuracy ($\hat{y}_t = y_t$) was for all considered shifts $> 85\%$, indicating that aligning via optimal transport is capable of apprehending the considered transformations. To confirm our findings, we also evaluated ROAR-S and the results demonstrate that the performance change caused by the shift is on average best mitigated when dropping features according to XPE.

Deriving intuitive and actionable insights Finally, we would like to demonstrate how the results obtained via XPE can yield novel and actionable insight about the model behavior under distribution shifts. To do so, we locally examine shift attributions on MNIST for digits where a certain shift had a particularly harmful effect on the prediction and seek to understand the reasons. In Figure 2, we plot some of these examples and notice that for such instances, the shifts do indeed perturb essential image regions in a way that suggests a different class. By consulting the different shift attribution results to narrow down a concrete reason we see that only XPE highlights the parts of the corruption that actually alter the appearance of a digit towards a different one. This indicates that the model is mainly misled by the intuitive regions, which cannot be concluded from the other results. Such kind of information can facilitate end-users to take efficient and targeted corrective measures for their application. Those can, for example, be given by cleaning or repairing the camera lens or removing ambient light sources causing harmful shadowing in important areas of the image.



(a) Zigzag perturbation

(b) Spatter perturbation

Figure 2: Feature shift importance on MNIST. The local explanations by XPE intuitively show the most convincing explanations only highlighting the part of the zigzag (a) connecting the top part of the '4', which changes the model's prediction to a '9'. A similar observation can be made for the spatter (b) corruption changing the prediction from '9' → '7'.

4 Conclusion

We introduced Explanatory Performance Estimation (XPE) as a novel framework to attribute an anticipated change in model performance induced by a distribution shift to individual features. Our approach requires no ground truth labels in the shifted domain, which corresponds to the typical situation in practice. Furthermore, we demonstrate the empirical success of our method through experiments, indicating that XPE can pave the way toward a more human-centered perspective to monitor ML-systems which is in line with the discussion in [37]. By this, the introduced method can facilitate a more efficient and targeted maintenance of black-box systems.

We anticipate the introduction of explainability in the monitoring of ML models as a fruitful research direction, e.g., combining XPE with concept-level explanations [46] can provide even more informative and actionable insights to reestablish reliable ML-systems.

References

[1] Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. In *6th International Conference on Learning Representations (ICLR)*, 2018.

[2] Robert W Batterman and Collin C Rice. Minimal model explanations. *Philosophy of Science*, 81(3):349–376, 2014.

[3] Umang Bhatt, Adrian Weller, and José MF Moura. Evaluating and aggregating feature-based model explanations. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 3016–3022, 2021.

[4] Kailash Budhathoki, Dominik Janzing, Patrick Bloebaum, and Hoiyi Ng. Why did the distribution change? In Arindam Banerjee and Kenji Fukumizu, editors, *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 1666–1674. PMLR, 13–15 Apr 2021.

[5] Nicolas Courty, Rémi Flamary, Amaury Habrard, and Alain Rakotomamonjy. Joint distribution optimal transportation for domain adaptation. *Advances in Neural Information Processing Systems*, 30, 2017.

[6] Nicolas Courty, Rémi Flamary, and Devis Tuia. Domain adaptation with regularized optimal transport. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 274–289. Springer, 2014.

[7] Nicolas Courty, Rémi Flamary, Devis Tuia, and Alain Rakotomamonjy. Optimal transport for domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(9):1853–1865, 2016.

[8] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

[9] Thomas Decker, Ralf Gross, Alexander Koebler, Michael Lebacher, Ronald Schnitzer, and Stefan H Weber. The thousand faces of explainable ai along the machine learning life cycle: Industrial reality and current state of research. In *International Conference on Human-Computer Interaction*, pages 184–208. Springer, 2023.

[10] Rémi Flamary, Nicolas Courty, Alexandre Gramfort, Mokhtar Z. Alaya, Aurélie Boisbunon, Stanislas Chambon, Laetitia Chapel, Adrien Corenflos, Kilian Fatras, Nemo Fournier, Léo Gautheron, Nathalie T.H. Gayraud, Hicham Janati, Alain Rakotomamonjy, Ievgen Redko, Antoine Rolet, Antony Schutz, Vivien Seguy, Danica J. Sutherland, Romain Tavenard, Alexander Tong, and Titouan Vayer. Pot: Python optimal transport. *Journal of Machine Learning Research*, 22(78):1–8, 2021.

[11] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM computing surveys (CSUR)*, 46(4):1–37, 2014.

[12] Rosana Noronha Gemaque, Albert França Josuá Costa, Rafael Giusti, and Eulanda Miranda Dos Santos. An overview of unsupervised drift detection methods. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 10(6):e1381, 2020.

[13] Anna Hedström, Leander Weber, Daniel Krakowczyk, Dilyara Bareeva, Franz Motzkus, Wojciech Samek, Sebastian Lapuschkin, and Marina M-C Höhne. Quantus: An explainable ai toolkit for responsible evaluation of neural network explanations and beyond. *Journal of Machine Learning Research*, 24(34):1–11, 2023.

[14] Sara Hooker, Dumitru Erhan, Pieter-Jan Kindermans, and Been Kim. A benchmark for interpretability methods in deep neural networks. *Advances in neural information processing systems*, 32, 2019.

[15] Hicham Janati, Marco Cuturi, and Alexandre Gramfort. Spatio-temporal alignments: Optimal transport through space and time. In *International Conference on Artificial Intelligence and Statistics*, pages 1695–1704. PMLR, 2020.

[16] Krishnaram Kenthapadi, Himabindu Lakkaraju, Pradeep Natarajan, and Mehrnoosh Sameki. Model monitoring in practice: Lessons learned and open challenges. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4800–4801, 2022.

[17] Ashkan Khakzar, Soroosh Baselizadeh, Saurabh Khanduja, Christian Rupprecht, Seong Tae Kim, and Nassir Navab. Neural response interpretation through the lens of critical pathways. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13528–13538, 2021.

[18] Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. From word embeddings to document distances. In *International conference on machine learning*, pages 957–966. PMLR, 2015.

[19] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998.

[20] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[21] Yanbin Liu, Linchao Zhu, Makoto Yamada, and Yi Yang. Semantic correspondence as an optimal transport problem. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4463–4472, 2020.

[22] Jie Lu, Anjin Liu, Fan Dong, Feng Gu, Joao Gama, and Guangquan Zhang. Learning under concept drift: A review. *IEEE transactions on knowledge and data engineering*, 31(12):2346–2363, 2018.

[23] Scott M Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. From local explanations to global understanding with explainable ai for trees. *Nature machine intelligence*, 2(1):56–67, 2020.

[24] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

[25] Sasu Mäkinen, Henrik Skogström, Eero Laaksonen, and Tommi Mikkonen. Who needs mlops: What data scientists seek to accomplish and how can mlops help? In *2021 IEEE/ACM 1st Workshop on AI Engineering-Software Engineering for AI (WAIN)*, pages 109–112. IEEE, 2021.

[26] Chenlin Meng, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021.

[27] Carlos Mougan, Klaus Broelemann, Gjergji Kasneci, Thanassis Tiropanis, and Steffen Staab. Explanation shift: Detecting distribution shifts on tabular data via the explanation space. In *NeurIPS 2022 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2022.

[28] Norman Mu and Justin Gilmer. Mnist-c: A robustness benchmark for computer vision. *ArXiv*, abs/1906.02337, 2019.

[29] David Nigenda, Zohar Karnin, Muhammad Bilal Zafar, Raghu Ramesha, Alan Tan, Michele Donini, and Krishnaram Kenthapadi. Amazon sagemaker model monitor: A system for real-time insights into deployed machine learning models. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, page 3671–3681, New York, NY, USA, 2022. Association for Computing Machinery.

[30] Andrei Paleyes, Raoul-Gabriel Urma, and Neil D Lawrence. Challenges in deploying machine learning: a survey of case studies. *ACM Computing Surveys*, 55(6):1–29, 2022.

[31] Yingxue Pang, Jianxin Lin, Tao Qin, and Zhibo Chen. Image-to-image translation: Methods and applications. *IEEE Transactions on Multimedia*, 2021.

[32] George Papamakarios, Eric T Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *J. Mach. Learn. Res.*, 22(57):1–64, 2021.

[33] Gabriel Peyré, Marco Cuturi, et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

[34] Joaquin Quinonero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D Lawrence. *Dataset shift in machine learning*. Mit Press, 2008.

[35] Stephan Rabanser, Stephan Günnemann, and Zachary Lipton. Failing loudly: An empirical study of methods for detecting dataset shift. *Advances in Neural Information Processing Systems*, 32, 2019.

[36] David Sculley, Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips, Dietmar Ebner, Vinay Chaudhary, Michael Young, Jean-Francois Crespo, and Dan Dennison. Hidden technical debt in machine learning systems. *Advances in neural information processing systems*, 28, 2015.

[37] Murtuza N Shergadwala, Himabindu Lakkaraju, and Krishnaram Kenthapadi. A human-centric perspective on model monitoring. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, volume 10, pages 173–183, 2022.

[38] Stefan Studer, Thanh Binh Bui, Christian Drescher, Alexander Hanuschkin, Ludwig Winkler, Steven Peters, and Klaus-Robert Müller. Towards crisp-ml (q): a machine learning process model with quality assurance methodology. *Machine learning and knowledge extraction*, 3(2):392–413, 2021.

[39] Mukund Sundararajan and Amir Najmi. The many shapley values for model explanation. In *International conference on machine learning*, pages 9269–9278. PMLR, 2020.

[40] Ankur Taly, Kaz Sato, and Claudiu Gruia. Monitoring feature attributions: How google saved one of the largest ml services in trouble. *Google Cloud Blog*, 2021.

[41] Arnaud Van Looveren, Janis Klaise, Giovanni Vacanti, Oliver Cobb, Ashley Scillitoe, Robert Samoilescu, and Alex Athorne. Alibi detect: Algorithms for outlier, adversarial and drift detection, 2019.

[42] Cédric Villani. *Optimal transport: old and new*, volume 338. Springer, 2009.

[43] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[44] Jiancheng Yang, Rui Shi, Donglai Wei, Zequan Liu, Lin Zhao, Bilian Ke, Hanspeter Pfister, and Bingbing Ni. Medmnist v2-a large-scale lightweight benchmark for 2d and 3d biomedical image classification. *Scientific Data*, 10(1):41, 2023.

[45] Chih-Kuan Yeh, Cheng-Yu Hsieh, Arun Suggala, David I Inouye, and Pradeep K Ravikumar. On the (in) fidelity and sensitivity of explanations. *Advances in Neural Information Processing Systems*, 32, 2019.

[46] Chih-Kuan Yeh, Been Kim, and Pradeep Ravikumar. Human-centered concept explanations for neural networks. In *Neuro-Symbolic Artificial Intelligence: The State of the Art*, pages 337–352. IOS Press, 2022.

[47] Haoran Zhang, Harvineet Singh, and Shalmali Joshi. "why did the model fail?": Attributing model performance changes to distribution shifts. In *ICML 2022: Workshop on Spurious Correlations, Invariance and Stability*, 2022.

A Appendix

Following supplementary material will be presented to provide further details about the implementation and the proposed metrics.

A.1 Details on Experiments

In this section, we document more detail regarding the conducted experiments.

A.1.1 Data and Model Details

During our experiments, we considered the MNIST [19] and FashionMNSIT [43] dataset loaded directly from torchvision. We also considered two datasets include in the MedMNISTv2 benchmark [44], namely OrganaMNIST and PneumoniaMNIST shown in Figure 3. The datasets have been loaded using the Python API provided by the medmnist package. For each dataset we trained and evaluated a LeNet model [20]. The model has been trained for 100 epochs with early stopping based on a patience of 10 epochs. The training has been performed using PyTorch's Adam optimizer with a batch size of 16 and a learning rate of 1e-3. For all datasets we used corruptions proposed by [28], which are shown in Figure 4.

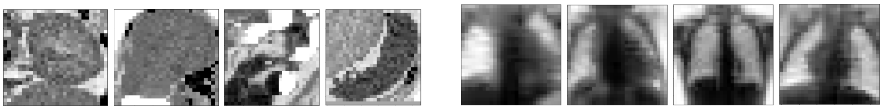


Figure 3: Medical images of organs and pneumonia used in the experiments in Table 1 (organaM and pneumM).

A.1.2 Shift Attribution Details

To estimate the couplings that align source and target samples we leveraged the POT library for optimal transport provided by [10]. All couplings are computed using linear programming based on the EMDTransport solver with default parameters including the squared Euclidean distance as a cost function.
To compute the necessary Shapley values for each shift attribution method relied on the model-agnostic KernelSHAP implementation provided by [24]. All Shapley values have been computed using a sample size of 3000 and the default choices for all other hyperparameters. For the AxS metric, we estimated the shift mask M^{KS} using a feature-wise two-sided Kolmogorov-Smirnov test with a 95% confidence threshold to signal a shift. The test statistic and the mask are computed using the implementation provided by [41].

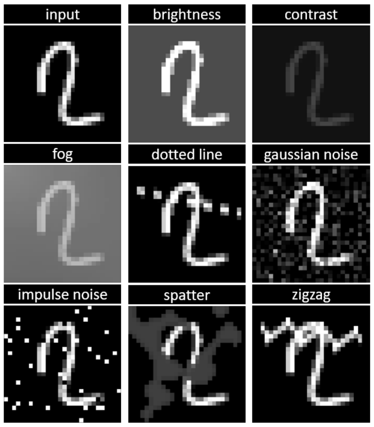


Figure 4: Example MNIST image illustrating the used corruptions

A.2 Metric details

A.2.1 Shift-Faithfulness and Complexity

For both metrics we randomly selected a subset of 500 samples from the test set of each data source and estimated all shift attributions as specified above. To compute the metrics we relied on the implementation provided by [13]. For the Shift-Faithfulness, we used the Faithfulness Correlation metric, specified the perturbation baseline to be the estimated per-shift version of each sample and chose a sample size of 100 with a subset size $|K|$ of 64. All other hyperparameters correspond to their default choices. Moreover, during the computation of such metrics, we removed samples where the anticipated performance change is only marginal as this causes either all shift attributions to be zero or causes numerical problems during the computation of the correlation value used in Shift-Faithfulness.

A.2.2 ROAR-S

For implementing the suggested remove, retrain, and shift (ROAR-S) metric we adapted the PyTorch implementation by [17] of the original remove and retrain (ROAR) metric introduced by [14].
Given the high computational requirements of that evaluation, we sub-sample the original training and test sets to obtain D_s^{train} and D_s^{test} each containing $N_{\textsf{ROAR}} = 1000$ samples. D_s^{train} is used to train the pre-removal LeNet model f. For each considered shift we created the corresponding shifted dataset versions D_t^{train} and D_t^{test} and computed for all elements x_t in the shifted train and test set the different feature shift attributions. For each attribution result $\phi^{\textsf{Shift}}$ we rank the individual importance values to obtain an ordered set. Afterward, we remove the top 5% of the features attributed the highest feature shift importance. If feature shift importance is attributed to less than 5% of the overall pixels, all those are removed. In the case of MNIST and FashionMNIST removing is performed by setting the value of the pixel to zero equalling the background value. For the MedMNIST datasets without a defined background value, the value of removed pixels is set to the mean of the respective dataset. This yields the post-removal sets $\tilde{D}_s^{train}$, $\tilde{D}_s^{test}$ and $\tilde{D}_t^{train}$ and $\tilde{D}_t^{test}$. The new source training $\tilde{D}_s^{train}$ is then used to retrain the LeNet model yielding $\tilde{f}$ and this new model is then evaluated on the post-removal test set to obtain the new performance decrease after removal. More specifically, based on the cross-entropy loss $\mathcal{L}$, the test performances of the original model f without removal are given by:

$$\mathcal{L}_t^{test} = \frac{1}{N_{\textsf{ROAR}}} \sum_{(x,y)\in D_t^{test}} \mathcal{L}(f(x), y)$$

$$\mathcal{L}_s^{test} = \frac{1}{N_{\textsf{ROAR}}} \sum_{(x,y)\in D_s^{test}} \mathcal{L}(f(x), y)$$

and the test performances after removal on the retrained model $\tilde{f}$ are given by:

$$\tilde{\mathcal{L}}_t^{test} = \frac{1}{N_{\textsf{ROAR}}} \sum_{(x,y)\in \tilde{D}_t^{test}} \mathcal{L}(\tilde{f}(x), y)$$

$$\tilde{\mathcal{L}}_s^{test} = \frac{1}{N_{\textsf{ROAR}}} \sum_{(x,y)\in \tilde{D}_s^{test}} \mathcal{L}(\tilde{f}(x), y)$$

Then, ROAR-S score can be defined as the proportion of performance decrease that remains after removal:

$$\textsf{ROAR-S}(\tilde{\mathcal{L}}_t^{test}, \tilde{\mathcal{L}}_s^{test}, \mathcal{L}_t^{test}, \mathcal{L}_s^{test}) = \frac{\max(0, \tilde{\mathcal{L}}_t^{test} - \tilde{\mathcal{L}}_s^{test})}{\mathcal{L}_t^{test} - \mathcal{L}_s^{test}}$$

For cases where no negative effect of the shift can be observed after removal, the metric is set to zero as the entire performance decrease has been mitigated. All models have been trained for 100 epochs with early stopping based on a patience of 10 epochs. The training has been performed using PyTorch's Adam optimizer with a batch size of 16 and a learning rate of 1e-2.