From Must to May: Enabling Test-Time Feature Imputation and Interventions

Evan Rex, Mateo Espinosa Zarlenga, Andrei Margeloiu and Mateja Jamnik

Department of Computer Science and Technology, University of Cambridge, United Kingdom

Abstract

Interpretable machine learning models can be improved by correcting mispredicted intermediate steps via *test-time interventions* on their intermediate predictions. Methods that jointly learn to impute missing features and predict a downstream task can benefit from such interventions. However, determining which features to prioritise for intervention remains a challenge. To address this, we propose *F-Act*, a novel method employing feature selection to adaptively manage feature availability during test-time. Our approach achieves this by combining in-model imputation and test-time interventions on intermediate predictions to avoid the need for model retraining. Furthermore, F-Act can recommend which features to prioritise when collecting data, a key property when optimising performance in resource-limited environments. Our empirical analysis shows F-Act performs competitively or better than previous baselines in inference tasks with missing features when incorporating feature collection recommendations. Additionally, we show F-Act can incorporate missing feature values through test-time interventions, improving predictive performance without retraining across tasks.

Keywords

Test-time interventions, Missing value imputation, Feature selection

1. Introduction

Machine learning models for tabular datasets typically expect a complete feature set during both training and inference. However, in practice, features are often missing during inference due to the high cost and difficulty of obtaining the complete feature set for some samples (e.g., gene expression counts) [1, 2]. Such test-time feature unavailability necessitates models that can make accurate predictions with incomplete feature sets. Additionally, since acquiring new features may be prohibitively expensive, it is crucial for these models to offer recommendations on which missing feature values to collect to maximise their impact on the model's accuracy.

Current strategies addressing limited feature availability typically involve either: (i) *imputing*, or predicting, missing features at test-time [1], or (ii) *selecting* a minimal feature subset on which the model is retrained [3, 4]. Although these methods are practical, they have clear limitations in scenarios with variable feature availability: Feature selection identifies critical features but cannot adapt to changes in feature availability, while imputation provides flexibility but lacks guidance for users on prioritising features.

In this paper, we address this gap by introducing *F-Act* (**F**eature-wise **Act**ive adaptation), a method that combines feature selection with imputation to enable adaptation to variable feature

HI-AI@KDD, Human-Interpretable AI Workshop at the KDD 2024, 26th of August 2024, Barcelona, Spain

er647@cam.ac.uk (E. Rex); me466@cam.ac.uk (M. E. Zarlenga); am2770@cam.ac.uk (A. Margeloiu); mj201@cam.ac.uk (M. Jamnik)

availability without retraining, all while maintaining high predictive accuracy. F-Act achieves this by, first, imputing missing features at test-time and, second, enabling new features to be incorporated through *test-time interventions*, where F-Act's intermediate predictions space is modified to incorporate the presence of a new feature. Technically, F-Act employs differentiable mask sampling and feature reconstruction to learn to optimally operate from an incomplete set of features. This design enables F-Act to advise on the order features should be collected to maximise their impact, permitting deployment in resource-constrained settings. Using real and synthetic datasets, we evaluate F-Act and find that it matches benchmarks' performance in imputation, feature selection, and prediction, providing recommendations that enhance model performance through adaptive feature incorporation.

2. Background and Related Work

Imputation and Feature Selection Our work incorporates both feature imputation and feature selection to address limited test-time feature availability. As such, our work is placed at the intersection of these two research subfields. Previous works in feature imputation can be divided into auxiliary model-based approaches and joint learning approaches. In this context, auxiliary-model-based approaches pair prediction models with separate imputation methods [5, 6, 7] while joint learning approaches integrate both prediction and feature selection in an end-to-end model [8, 9]. Nevertheless, we emphasise that previously proposed imputation techniques lack feature prioritisation for collection, leading to uncertainty over what features one should prioritise when deploying the model in a setup with varying feature availability. This is a key gap we aim to address with this paper.

Feature selection techniques [10, 11, 12, 13], in contrast, deal with potential feature unavailability (or redundancy) by learning to select a subset of features from which a task can be accurately solved. These approaches commonly achieve this by learning a feature importance ranking that can then inform which subset of features one should select. A shortcoming of these approaches, however, is their inflexibility to a varying set of input features, as, once features have been selected, they require a *fixed* subset of features to train the downstream model [14]. As such, in this work we combine feature imputation with feature selection to enable easy adaptability from a core set of initially selected features. We note that combining feature selection and imputation has been previously explored [15, 16, 17, 3]. However, performing feature selection with joint learning for inference and imputation in a single end-to-end architecture is novel. This is worth exploring, as Bertsimas et al. [8] and Le Morvan et al. [9] note that joint learning for imputation and inference can yield improved results.

Relation to Active Feature Acquisition Active Feature Acquisition (AFA) involves learning a policy for collecting new features at test-time such that a model's accuracy is maximised after observing a small set of features [18, 19, 20, 21]. As we are interested in providing feature collection recommendations, our work is highly related to AFA. Nevertheless, we highlight that we distinguish ourselves from traditional AFA approaches in two key ways. First, we provide feature collection recommendations at a global level rather than at a local, per-sample level. Second, we learn to both select a subset of features and impute missing features in an end-to-end

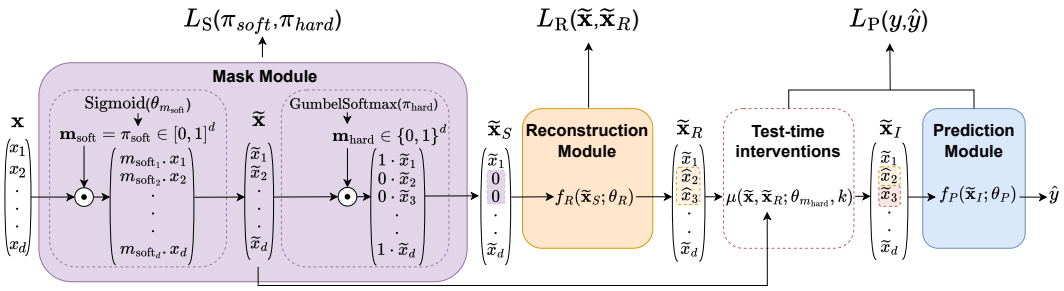


Figure 1: F-ACT. Given a sample $\mathbf{x}$, we apply an element-wise mask $\mathbf{m}_{\text{soft}} \in [0,1]^n$, which acts as a sparsity-inducing global feature selection mask, eliminating noisy features and resulting in $\tilde{\mathbf{x}}$. We then apply an element-wise mask $\mathbf{m}_{\text{hard}} \in \{0,1\}^n$, sampled from a Gumbel-Softmax distribution with learnable probabilities π_{hard}. The masked features are reconstructed by the Reconstruction Module, f_R, to approximate $\tilde{\mathbf{x}}$. Subsequently, we perform test-time interventions on this reconstructed sample by re-incorporating k previously masked features according to a greedy policy, μ derived from the rankings of π_{hard}, resulting in $\tilde{\mathbf{x}}_I$. Finally, the Prediction Module, f_P predicts the sample's label.

fashion, enabling missing features to be predicted at test-time.

Relation to Human Interpretable Artificial Intelligence Human Interpretable Artificial Intelligence (HI-AI) refers to AI systems designed to ensure their decisions and workings are understandable and transparent to humans. Our work is related to HI-AI methods as it enables (1) reconstruction of missing features through test-time imputations, providing insights into a model's understanding of how missing features relate to provided ones, (2) construction of feature importance rankings through its feature collection recommendations, and (3) test-time interventions, where users can provide previously missing features by intervening on F-ACT's intermediate predictions using these features' values. As such, our work is related to previous interpretable imputation techniques [22, 7] and methods in the concept-based explainable AI literature [23, 24, 25, 26] that provide test-time feedback to models via human-aligned concepts.

3. Feature-wise Active adaptation

We present a joint learning framework for inference and missing value imputation that offers users insights into which missing feature values should be prioritised for collection and intervention. Formally, our goal is to learn a predictor f_θ, parameterised by θ, which can operate on any subset of features $S \subseteq \mathcal{F}$. Concurrently, the predictor f_θ should also suggest which missing feature values to collect for intervention at test-time, prioritised by their importance to improve the predictor's performance. We achieve this by introducing F-ACT (Figure 1), a method for the joint learning of feature selection, missing value imputation, and prediction. Our architecture comprises three modules: (i) a *Mask module* that facilitates feature selection, (ii) a *Reconstruction module* for feature imputation, and (iii) a *Prediction module* for making predictions.

The Mask module serves three objectives: i) global feature selection to eliminate irrelevant features, ii) learning feature importance rankings to provide recommendations for feature

collection, and iii) simulating a missing feature scenario to train the Reconstruction module. We achieve all this functionality through hierarchical masking, first employing a *soft mask* $\mathbf{m}_{\text{soft}} \in [0,1]^d$ for feature selection and then a *hard mask* $\mathbf{m}_{\text{hard}} \in \{0,1\}^d$ to simulate missing features. The hard mask is sampled from a Gumbel-Softmax distribution [27] with a learnable probability π_{hard}.

To enable predictions from any truncated feature space and facilitate test-time interventions, we use the Reconstruction Module to reconstruct features from the truncated feature space $\tilde{\mathcal{X}}_S$ generated by the hard mask. The Reconstruction Module outputs "reconstructed" samples, containing feature values for all values, even though they are missing from the original input. The reconstructed samples are then processed by the Prediction Module, which maps from the complete feature space $\tilde{\mathcal{X}}$ to make the predictions $\mathcal{Y}$. This setup ensures the model can make predictions even when in the presence of missing features.

To provide feature collection recommendations, we define a greedy intervention policy that corrects reconstructed data based on feature selection probabilities from the mask module. This is the same approach used by feature importance-based selection methods [11, 4, 13, 12].

In order to jointly learn to perform feature selection, missing feature imputation and, downstream task prediction, we train our model using a composite loss function $\mathcal{L} = \mathcal{L}_{\text{P}} + \alpha_S \mathcal{L}_{\text{S}} + \alpha_R \mathcal{L}_{\text{R}}$ where α_S and α_R are hyperparameters controlling how much we value feature selection (i.e., $\mathcal{L}_{\text{S}}$) and feature reconstruction (i.e., $\mathcal{L}_{\text{R}}$) over task accuracy (i.e., $\mathcal{L}_{\text{P}}$).

To encourage our model to perform a sparse feature selection, we follow previous works [28, 4] and let $\mathcal{L}_{\text{S}}$ be the ℓ_1 norm of the soft and hard learnable mask probabilities:

$$\mathcal{L}_{\text{S}} := \sum_{i=1}^{d} \left(\pi_{\text{soft}_i} + \pi_{\text{hard}_i} \right)$$

In contrast, to encourage accurate imputation of masked features, we include a reconstruction loss term $\mathcal{L}_{\text{R}}$ that minimises the ℓ_2 norm of the difference between reconstructed/imputed feature values and ground truth feature values:

$$\mathcal{L}_{\text{R}} := \frac{1}{|\mathcal{F} \setminus S|} \sum_{i \in \mathcal{F} \setminus S} \left(\tilde{\mathbf{x}}_i - f_{R_i}(\mathbf{m}_{\text{hard}} \odot \tilde{\mathbf{x}}; \theta_R) \right)^2$$

where S is the set of features selected by the Mask module. This loss encourages our model to learn to select a set of *core features* from which other, *dependent features*, may be easily imputed.

Finally, to enable our model to predict downstream tasks both in and outside the presence of potential feature interventions, we follow the work in [25] and define our prediction loss, $\mathcal{L}_{\text{P}}$, as

$$\mathcal{L}_{\text{P}} := \; L_{\text{pred}}(f_\theta(\mathbf{x}; \theta, 0), \mathbf{X}, \mathbf{Y}) + \; \omega^{k_{\text{max}}} L_{\text{pred}}(f_\theta(\mathbf{x}; \theta, k_{\text{max}}), \mathbf{X}, \mathbf{Y})$$

Here, $f_\theta(\mathbf{x}; \theta, i)$ represents the output of the task predictor for sample $\mathbf{x}$ when the top-i dependent features are intervened on and k_{max} is the maximum number of interventions one may perform (i.e., the number of dependent features $|\mathcal{F} \setminus S|$). We clarify that, in this context, an *intervention* for a feature $\mathbf{x}_j$ involves setting the j-th feature of the reconstructed features $\mathbf{x}_R$ to $\tilde{\mathbf{x}}_j$. This loss, therefore, encourages the model to minimise a task-specific loss (e.g., cross-entropy) before and after interventions, with higher penalties incurred when a mistake is made after a higher number of features have been intervened on at train time (controlled by a hyperparameter $\omega > 1$).

Table 1
Test F1 Scores (%) (class-weighted) are presented as the mean ± standard deviation over three seeds. To aggregate the results, we compute the average rank of each method across datasets, where a higher rank indicates superior accuracy. F-Act achieves competitive accuracy and overall ranks higher than other benchmark methods.

Model	COIL20	Isolet	PBMC	USPS	Finance	Madelon	Mice Protein	Avg. Rank
Lasso	98.24 ± 0.00	**94.58 ± 0.01**	89.25 ± 0.14	93.36 ± 0.03	59.78 ± 0.00	51.53 ± 0.00	95.24 ± 2.29	4.29
Rand. Forest	96.76 ± 0.18	90.09 ± 0.45	88.66 ± 0.37	93.36 ± 0.15	**61.95 ± 0.47**	67.19 ± 0.26	96.98 ± 1.82	4.07
XGBoost	98.61 ± 0.00	88.75 ± 0.00	89.42 ± 0.00	**97.37 ± 0.00**	58.83 ± 0.00	**80.96 ± 0.00**	98.15 ± 0.81	3.07
SEFS	94.97 ± 1.40	88.61 ± 2.08	83.17 ± 1.27	92.54 ± 0.75	59.93 ± 0.46	65.23 ± 1.69	85.08 ± 4.59	5.71
CAE	97.04 ± 0.87	80.14 ± 1.28	68.07 ± 5.34	90.47 ± 0.83	59.26 ± 1.2	70.19 ± 1.83	85.35 ± 5.37	5.86
Sup. CAE	6.46 ± 4.45	3.68 ± 0.79	85.37 ± 0.01	20.90 ± 2.56	54.44 ± 1.80	61.84 ± 0.30	17.24 ± 6.78	7.43
MLP	98.83 ± 0.53	93.48 ± 1.29	89.42 ± 0.48	96.78 ± 0.15	57.02 ± 3.96	57.18 ± 0.89	98.30 ± 0.27	3.36
F-Act (Ours)	**98.84 ± 0.19**	92.86 ± 1.18	**89.87 ± 0.40**	95.95 ± 0.31	59.81 ± 1.90	72.9 ± 2.33	**98.46 ± 1.07**	**2.21**

Inference By thresholding mask probabilities, we can identify core necessary features and recommend which features to prioritize for collection. During inference, F-Act imputes missing features dynamically and allows the re-incorporation of non-selected features in the form of test-time interventions. In practice, given an incomplete sample at test-time, we replace the missing values with 0. To perform feature selection, imputation and prediction, we apply the mask, reconstruction and prediction modules in order. A change from the training procedure is that, at test-time, the Gumbel Softmax function's temperature is set to 0, making its performance deterministic. This is equivalent to thresholding the hard mask probabilities at 0.5. Test-time interventions are performed by replacing the reconstructions of the hard-masked features with their true values, as shown in Figure 1.

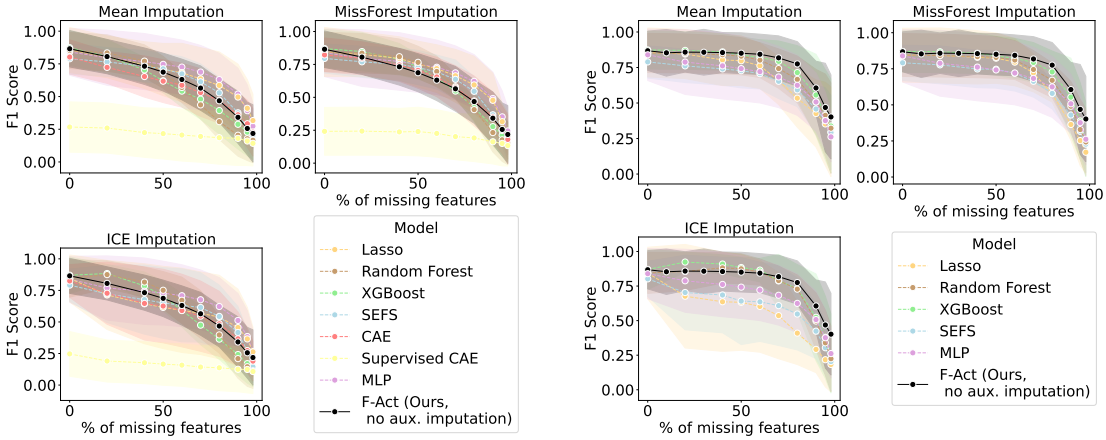
4. Experiments

Datasets and benchmark methods We consider various real-world datasets commonly referenced in feature selection literature. These include image datasets (COIL20 and USPS), a voice audio dataset (Isolet) sourced from [29], a synthetic dataset (Madelon) from [30], genomic datasets (PBMC [31] and Mice Protein [32]), and a financial dataset (Finance) from [33].

Beyond predictive accuracy, we assess F-Act's capabilities in selecting important features and recommending which features to collect at test-time to enhance performance. To this end, we consider several feature selection methods, including LASSO [11], Random Forest [13], Concrete Autoencoders (CAE) [34], XGBoost [12], and SEFS [4]. All methods except CAE rank the features by their importance, which allows us to evaluate F-Act's ability to recommend features for collection. We train each feature selection method on the prediction task, using a simple MLP as a baseline for comparison.

For missing data imputation, we evaluate four methods: Mean, Iterative Chained Equations (ICE) [6], and MissForest [7]. Additionally, we assess the performance of all combinations of downstream models and imputation methods.

We train F-Act by minimising the loss L. We pre-train the reconstruction module following [4], with tasks that include reconstructing input vectors and estimating gate vectors. Following this, we tune the intervention number k to minimise the prediction loss on the validation dataset.



(a) **Missing Completely at Random (MCAR)** (b) **Collecting features by model's feature ranking**

Figure 2: Performance varies with the number of missing features at test-time; error bars represent standard deviation. (a) When features are missing at random, several methods outperform F-Act. (b) F-Act outperforms other methods when the features selected for collection at test-time are based on the predictor's learned feature ranking. F-Act outperforms all baseline methods when many features are missing. However, with fewer missing features, F-Act can be outperformed by XGBoost combined with ICE. This indicates that F-Act is especially effective when feature collection can be prioritised.

For further implementation details, please refer to Appendix A.

Predictive Accuracy Table 1 illustrates the predictive accuracy of F-Act compared to other benchmark methods. F-Act demonstrates competitive performance, consistently ranking in the top three across datasets and outperforming all baselines in three cases. Overall, F-Act ranks the best across datasets. However, besides making predictions, F-Act offers two additional functionalities *without any re-training*: imputing missing data at test-time and recommending which feature values to collect. We next explore these capabilities.

Test-time Imputation First, we evaluate F-Act's imputation capabilities when features are missing completely at random (MCAR). To simulate this, we randomly remove features at test-time (without considering the potential to collect these feature values). At low levels of missing features, Figure 2a shows that F-Act is generally outperformed by Random Forest and MLP, and at higher levels of missing values, it is outperformed by Lasso and MLP. These mixed results suggest F-Act achieves relatively average performance when one cannot utilise its learned feature ranking.

Second, we consider prioritising test-time feature collection based on the model's learned feature ranking. Figure 2b shows that when this ranking is used, F-Act outperforms all other methods with only a few features collected.

Test-Time Interventions As a feature selection method, F-Act uses a threshold to separate core from non-core features. Unlike standard approaches, F-Act can incorporate non-core

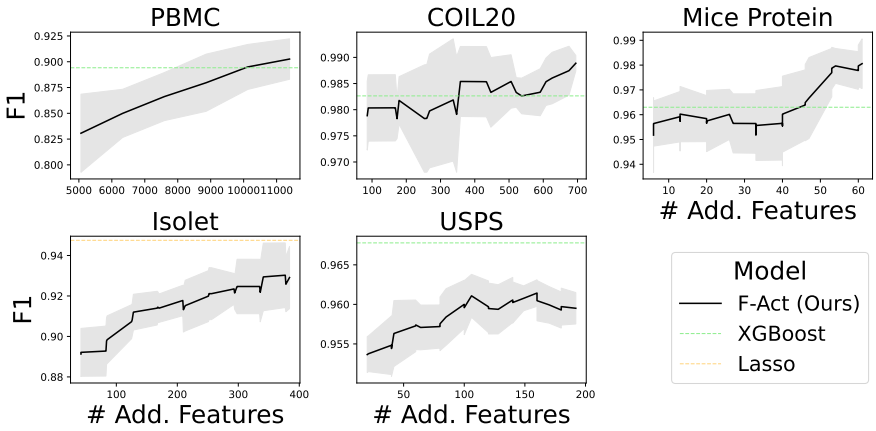


Figure 3: Test-time interventions boost performance. The x-axis represents the number of interventions. The dotted line marks the peak performance achieved by baselines trained with the full feature set, whereas F-Act operates with only a subset. The results demonstrate that F-Act's performance improves when values for the recommended features are imputed at test-time. In some cases, this enhancement allows F-Act, without retraining, to surpass baselines that need the full feature set at test-time.

features during inference. Figure 3 illustrates that F-Act's performance improves with test-time interventions, sometimes even surpassing the best-performing method on that dataset. For more results, please see Appendix B.

5. Conclusion

This paper introduces F-Act, a method combining feature selection and missing data imputation to enable the model to operate when there are missing features at test-time. More importantly, F-Act provides recommendations of which features one should prioritise collecting at test-time to improve the model's performance. Our empirical analysis shows F-Act performs competitively or better than previous baselines in inference tasks with missing features when incorporating feature collection recommendations. Additionally, we show how F-Act can incorporate missing feature values at test-time through test-time interventions, improving performance without retraining and boosting F1 scores across datasets. This work highlights the benefit of designing methods that learn, in an end-to-end fashion, to adapt to different feature availability while providing feature collection recommendations.

References

[1] M. H. A. Kohbalan Moorthy, C. W. H. Mohd Arfian Ismail, S. Mohd Saberi Mohamad, An evaluation of machine learning algorithms for missing values imputation, InternationalJournal of Innovative Technology and Exploring Engineering 8 (2019) 415–420.

[2] S. D. Grosse, J. M. Gudgeon, Cost or price of sequencing? implications for economic evaluations in genomic medicine, Genetics in Medicine 23 (2021) 1833–1835.

[3] J. Cai, L. Fan, X. Xu, X. Wu, Unsupervised and supervised feature selection for incomplete data via l2, 1-norm and reconstruction error minimization, Applied Sciences 12 (2022) 8752.

[4] C. Lee, F. Imrie, M. van der Schaar, Self-supervision enhanced feature selection with correlated gates, in: International Conference on Learning Representations, 2022.

[5] D. Jarrett, B. C. Cebere, T. Liu, A. Curth, M. van der Schaar, HyperImpute: Generalized iterative imputation with automatic model selection, in: K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, S. Sabato (Eds.), Proceedings of the 39th International Conference on Machine Learning, volume 162 of *Proceedings of Machine Learning Research*, PMLR, 2022, pp. 9916–9937. URL: https://proceedings.mlr.press/v162/jarrett22a.html.

[6] S. van Buuren, Multiple imputation of discrete and continuous data by fully conditional specification, Statistical Methods in Medical Research 16 (2007) 219–242.

[7] D. J. Stekhoven, P. Bühlmann, Missforest–non-parametric missing value imputation for mixed-type data, Bioinformatics 28 (2012) 112–118.

[8] D. Bertsimas, A. Delarue, J. Pauphilet, Beyond impute-then-regress: Adapting prediction to missing data, ArXiv preprint abs/2104.03158 (2021). URL: https://arxiv.org/abs/2104.03158.

[9] M. Le Morvan, J. Josse, T. Moreau, E. Scornet, G. Varoquaux, Neumiss networks: differentiable programming for supervised learning with missing values., Advances in Neural Information Processing Systems 33 (2020) 5980–5990.

[10] V. Bolón-Canedo, N. Sánchez-Maroño, A. Alonso-Betanzos, A review of feature selection methods on synthetic data, Knowledge and information systems 34 (2013) 483–519.

[11] R. Tibshirani, Regression shrinkage and selection via the lasso, Journal of the Royal Statistical Society, Series B 58 (1996) 267–288.

[12] T. Chen, C. Guestrin, Xgboost: A scalable tree boosting system, in: B. Krishnapuram, M. Shah, A. J. Smola, C. C. Aggarwal, D. Shen, R. Rastogi (Eds.), Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016, ACM, 2016, pp. 785–794. URL: https://doi.org/10.1145/2939672.2939785. doi:10.1145/2939672.2939785.

[13] L. Breiman, Random forests, Machine Learning 45 (2001) 5–32.

[14] I. C. Covert, W. Qiu, M. Lu, N. Y. Kim, N. J. White, S.-I. Lee, Learning to maximize mutual information for dynamic feature selection, in: A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, J. Scarlett (Eds.), Proceedings of the 40th International Conference on Machine Learning, volume 202 of *Proceedings of Machine Learning Research*, PMLR, 2023, pp. 6424–6447. URL: https://proceedings.mlr.press/v202/covert23a.html.

[15] A. M. Sefidian, N. Daneshpour, Missing value imputation using a novel grey based fuzzy c-means, mutual information based feature selection, and regression model, Expert Systems with Applications 115 (2019) 68–94. URL: https://www.sciencedirect.com/science/article/pii/S0957417418304822. doi:https://doi.org/10.1016/j.eswa.2018.07.057.

[16] G. Doquire, M. Verleysen, Feature selection with missing data using mutual information estimators, Neurocomputing 90 (2012) 3–11.

[17] P. Meesad, K. Hengpraprohm, Combination of knn-based feature selection and knnbased missing-value imputation of microarray data, in: 2008 3rd International Conference on Innovative Computing Information and Control, 2008, pp. 341–341. doi:10.1109/ICICIC.2008.635.

[18] M. Saar-Tsechansky, P. Melville, F. Provost, Active feature-value acquisition, Management Science 55 (2009) 664–684.
[19] Y. Li, J. Oliva, Active feature acquisition with generative surrogate models, in: M. Meila, T. Zhang (Eds.), Proceedings of the 38th International Conference on Machine Learning, volume 139 of *Proceedings of Machine Learning Research*, PMLR, 2021, pp. 6450–6459. URL: https://proceedings.mlr.press/v139/li21p.html.
[20] H. Shim, S. J. Hwang, E. Yang, Joint active feature acquisition and classification with variable-size set encoding, in: S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, R. Garnett (Eds.), Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada, 2018, pp. 1375–1385. URL: https://proceedings.neurips.cc/paper/2018/hash/e5841df2166dd424a57127423d276bbe-Abstract.html.
[21] Y. Li, J. Oliva, Active feature acquisition with generative surrogate models, in: International Conference on Machine Learning, PMLR, 2021, pp. 6450–6459.
[22] M. J. Azur, E. A. Stuart, C. Frangakis, P. J. Leaf, Multiple imputation by chained equations: what is it and how does it work?, International journal of methods in psychiatric research 20 (2011) 40–49.
[23] P. W. Koh, T. Nguyen, Y. S. Tang, S. Mussmann, E. Pierson, B. Kim, P. Liang, Concept bottleneck models, in: Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event, volume 119 of *Proceedings of Machine Learning Research*, PMLR, 2020, pp. 5338–5348. URL: http://proceedings.mlr.press/v119/koh20a.html.
[24] M. Espinosa Zarlenga, P. Barbiero, G. Ciravegna, G. Marra, F. Giannini, M. Diligenti, Z. Shams, F. Precioso, S. Melacci, A. Weller, et al., Concept embedding models: Beyond the accuracy-explainability trade-off, Advances in Neural Information Processing Systems 35 (2022) 21400–21413.
[25] M. Espinosa Zarlenga, K. Collins, K. Dvijotham, A. Weller, Z. Shams, M. Jamnik, Learning to receive help: Intervention-aware concept embedding models, Advances in Neural Information Processing Systems 36 (2024).
[26] R. Marcinkevičs, S. Laguna, M. Vandenhirtz, J. E. Vogt, Beyond concept bottleneck models: How to make black boxes intervenable?, arXiv preprint arXiv:2401.13544 (2024).
[27] E. Jang, S. Gu, B. Poole, Categorical reparameterization with gumbel-softmax, arXiv preprint arXiv:1611.01144 (2016).
[28] A. Margeloiu, N. Simidjievski, P. Lio, M. Jamnik, Weight predictor network with feature selection for small sample tabular biomedical data, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 37, 2023, pp. 9081–9089.
[29] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, H. Liu, Feature selection: A data perspective, ACM Computing Surveys (CSUR) 50 (2018) 94.
[30] I. Guyon, Madelon, UCI Machine Learning Repository, 2008. DOI: https://doi.org/10.24432/C5602H.
[31] A. Gayoso, Z. Steier, R. Lopez, J. Regier, K. L. Nazor, A. Streets, N. Yosef, Joint probabilistic modeling of paired transcriptome and proteome measurements in single cells, Biorxiv (2020) 2020–05.
[32] C. Higuera, K. Gardiner, K. Cios, Mice protein expression, UCI Machine Learning Reposi-

tory, 2015. DOI: https://doi.org/10.24432/C50S3Z.
[33] N. Carbone, 200+ financial indicators of us stocks (2014-2018), 2020. URL: https://www.kaggle.com/datasets/cnic92/200-financial-indicators-of-us-stocks-20142018.
[34] M. F. Balın, A. Abid, J. Zou, Concrete autoencoders: Differentiable feature selection and reconstruction, in: K. Chaudhuri, R. Salakhutdinov (Eds.), Proceedings of the 36th International Conference on Machine Learning, volume 97 of *Proceedings of Machine Learning Research*, PMLR, 2019, pp. 444–453. URL: https://proceedings.mlr.press/v97/balin19a.html.
[35] R. Salakhutdinov, Deep learning, in: S. A. Macskassy, C. Perlich, J. Leskovec, W. Wang, R. Ghani (Eds.), The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014, ACM, 2014, p. 1973. URL: https://doi.org/10.1145/2623330.2630809. doi:10.1145/2623330.2630809.
[36] I. R. White, P. Royston, A. M. Wood, Multiple imputation using chained equations: issues and guidance for practice, Statistics in medicine 30 (2011) 377–399.

A. Reproducibility

A.1. Datasets

Table 2
Overview of Datasets

Dataset	# samples (N)	# features (D)	# classes	N/D	min # samples per class	max # samples per class	Domain
COIL20 [29]	1440	1024	20	1	72	72	Image
Finance [33]	2664	154	2	17	1195	1469	Financial indicators
Isolet [29]	1560	617	26	3	60	60	Voice audio
Madelon [30]	2600	500	2	5	1300	1300	Synthetic
Mice Protein [32]	1080	77	8	14	105	150	Protein Expression
PBMC [31]	1038	21932	2	0	514	524	Genomics
USPS [29]	9298	256	10	36	708	1553	Image (written digit)

A.2. Reproducibility

Our code is made available at https://github.com/evanrex/feature-wise-active-adaptation.

A.3. Training Protocol

We present the training algorithm for our approach in Appendix A.3.

A.4. Training and Evaluation Methodology

We divided the datasets into three subsets: training, validation, and testing, using a 60:20:20 split. We use the validation to select the model's hyperparameters. We evaluate and report the class-weighted F1 score on the test set. The results are averaged across these seeds during both

Algorithm 1 Pre-training F-ACT

Require: Dataset $(\mathbf{X}, \mathbf{Y})$, mini-batch size n_{mb}, learning rate η, Loss coefficient α_{pre}, mask probability π

Ensure: Trained model parameters θ_R

Initialise parameters $\theta_R, \theta_{\text{pre}}$

repeat ▷ Begin pre-training loop

  for $i = 1$ to n_{mb} **do** ▷ For each point in the mini-batch

    $(\mathbf{x}_i, y_i) \sim (\mathbf{X}, \mathbf{Y})$ ▷ Sample a data point

    $\mathbf{m}_{\text{hard}_i} \leftarrow \text{Bernoulli}(\pi)$ ▷ Sample hard mask with probability π

    $\mathbf{x}_{S_i} \leftarrow \mathbf{m}_{\text{hard}_i} \odot \mathbf{x}_i$ ▷ Apply hard mask to the data point

    $\mathbf{x}_{R_i} \leftarrow f_R(\mathbf{x}_{S_i}; \theta_R)$ ▷ Reconstruct features from masked data

    $\hat{\mathbf{m}}_{\text{hard}_i} \leftarrow f_{\text{pre}}(\mathbf{x}_{S_i}; \theta_{\text{pre}})$ ▷ Predict mask using the pre-training function

  end for

  $\theta \leftarrow \theta - \eta \nabla_\theta \sum_{i=1}^{n_{\text{mb}}} \left(\mathcal{L}_{\text{R}}(\mathbf{x}_{S_i}, \mathbf{x}_{R_i}) + \alpha_{\text{pre}} \cdot \text{CE}(\hat{\mathbf{m}}_{\text{hard}_i}, \mathbf{m}_{\text{hard}_i}) \right)$ ▷ Update parameters using gradient descent

until convergence

hyperparameter tuning and final evaluation phases. We pre-train out model as per Appendix A.3 and train our model as per Algorithm 2.

A.5. Hyper-parameter Tuning

Random Forest. For the Random Forest model, we conducted a hyper-parameter sweep on the `max_depth` parameter. The values considered for `max_depth` were {3, 5, 7}. This tuning was performed to control the complexity of the individual trees in the forest, with a goal of balancing the bias-variance tradeoff.

Lasso. In our implementation of the Lasso model, we performed hyper-parameter tuning on two key parameters: `l1_ratio` and C. The `l1_ratio` was varied over {0, 0.25, 0.5, 0.75, 1}, allowing us to explore the impact of the ElasticNet mixing parameter which adjusts the balance between L1 and L2 penalties. The C parameter, which controls the inverse of regularisation strength, was swept over {10, 100, 1000}, providing a wide range of regularisation effects.

XGBoost. For the XGBoost model, we focused our hyper-parameter sweep on the `eta` (learning rate) and `max_depth`. The `eta` values considered were {0.1, 0.3, 0.5}, providing a spectrum of learning rates to control the step size during optimisation. For `max_depth`, the values were {3, 6, 9}, allowing us to examine different depths for the trees to manage the model's complexity and prevent overfitting.

Neural Network Based Models. For the neural network-based models, which include MLP, SEFS, CAE, Supervised CAE, and F-ACT, we conducted a hyper-parameter sweep. Key parameters included learning rate, `lr`, ({1e-3, 3e-4, 1e-4}), number of hidden layers ({1, 2, 4}), and `dropout` rates ({0, 0.2}). Additionally, for the Concrete Autoencoder models (CAE, Supervised CAE), we swept the `neurons_ratio` over {0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0} to explore various proportions of neurons in the encoder and decoder layers. This extensive tuning process was aimed at optimising each of the methods architectures and regularisation techniques to

Algorithm 2 Training F-ACT

Require: Dataset $(\mathbf{X}, \mathbf{Y})$, mini-batch size n_{mb}, loss coefficients (α_S, α_R), Gumbel-Softmax temperature parameter τ, maximum intervention k_{max}, learning rate η

Ensure: Trained model parameters $(\theta_{m_{\text{soft}}}, \theta_{m_{\text{hard}}}, \theta_R, \theta_P)$

Initialise $(\theta_{m_{\text{soft}}}, \theta_{m_{\text{hard}}}, \theta_R, \theta_P)$ ▷ Initialise parameter weights randomly
repeat
  for $i = 1$ to n **do** ▷ For each sample in the training set
    $(\mathbf{x}_i, y_i) \sim (\mathbf{X}, \mathbf{Y})$ ▷ Sample a data point
  Mask the data
    $\mathbf{m}_{\text{soft}} \leftarrow \text{Sigmoid}(\theta_{m_{\text{soft}}})$ ▷ Compute soft mask
    $\mathbf{m}_{\text{hard}} \leftarrow \text{GumbelSoftmax}(\theta_{m_{\text{hard}}}, \tau)$ ▷ Compute hard mask
    $\tilde{\mathbf{x}}_i \leftarrow \mathbf{m}_{\text{soft}} \odot \mathbf{x}_i$ ▷ Apply soft mask to input data
    $\tilde{\mathbf{x}}_{S_i} \leftarrow \mathbf{m}_{\text{hard}} \odot \tilde{\mathbf{x}}_i$ ▷ Apply hard mask to soft-masked data
  Reconstruct the data, apply interventions, and make prediction
    $\tilde{\mathbf{x}}_{R_i} \leftarrow f_R(\tilde{\mathbf{x}}_{S_i}; \theta_R)$ ▷ Reconstruct the hard-masked features
    $\hat{y}_{R_i} \leftarrow f_P(\tilde{\mathbf{x}}_{R_i}; \theta_P)$ ▷ Make prediction without intervention
    $\tilde{\mathbf{x}}_{I_i} \leftarrow \mu(\tilde{\mathbf{x}}_{R_i}, \tilde{\mathbf{x}}_i; \theta_{m_{\text{hard}}}, k_{\text{max}})$ ▷ Apply interventions on top features
    $\hat{y}_{I_i} \leftarrow f_P(\tilde{\mathbf{x}}_{I_i}; \theta_P)$ ▷ Make prediction with full intervention
  end for
  $\theta \leftarrow \theta - \eta \nabla_\theta \sum_{i=1}^{n_{\text{b}}} \left(\mathcal{L}(\hat{y}_{R_i}, \hat{y}_{I_i}, \mathbf{y}_i) \right)$ ▷ Update parameters using gradient descent
until convergence

enhance model performance

B. Further Experiments and Discussion

B.1. Test-time Imputation

B.1.1. Further Discussion

In Figure 2b, we observe the interesting phenomenon that at low levels of missing data, ICE imputation enables the tree-based models to achieve considerably improved performance. Concurrently, ICE negatively affects the Neural Network and Lasso models. Here, we discuss that phenomenon. Note that, as per the structure of the experiment, the missing features at those levels are relatively lower-ranked features which have less impact on predictions in the case of tree-based models. One possible explanation for the increase in performance of the tree-based models, is that they were initially negatively affected by over-fitting to noise in the lower-ranked features. As a result, the models might benefit from the removal of these features in the test set. This then explains why replacing the lower-ranked features with expected values conditioned by the other observed features of that data point (as per the ICE imputation strategy [22]) would reduce their noisy impact. To explain the poor performance of ICE with the Neural Network-based and Lasso models, we note that those models are known to be more sensitive to scale and domain shift [35]. We also note that ICE is known to suffer from misspecification

Table 3
F1 Weighted Averages. Comparing F-Act, which uses an "optimal feature selection" found through post-training hyperparameter tuning of the number of interventions, to F-Act "selected only", which uses a feature selection made from thresholding the feature selection probabilities as 50%.

	Partition	Validation	Validation	Test	Test
	Model	F-Act (selected only)	F-Act	F-Act (selected only)	F-Act
Dataset	COIL20	0.9929	**0.9953**	0.9860	**0.9884**
	Isolet	0.9264	**0.9430**	0.9197	**0.9286**
	PBMC	0.8461	**0.8793**	0.8317	0.8987
	USPS	**0.9681**	**0.9681**	**0.9683**	**0.9683**
	Finance	0.5992	**0.6027**	**0.6029**	0.5981
	Madelon	0.7287	**0.7415**	0.7225	**0.7290**
	Mice Protein	0.9815	**0.9908**	**0.9877**	0.9846

[36], where imputed values are "implausible", falling out of the domain.

B.2. Optimal feature availability

The adaptive nature of our model enables the ability to provide a more finely tuned optimal feature selection recommendation than the standard "selected" vs "non-selected" features dichotomy. Rather than being derived from what is typically an arbitrarily set threshold, we are able to tune the number of selected features *without re-training*. By contrast, to implement feature selection with methods such as Lasso it is standard to re-train the model with the reduced feature set. In this section, we hypothesise that this functionality will enable improved performance, due to the ability to more finely tune the number of selected features.

The "optimal feature selection" is found through post-training hyperparameter tuning of the number of interventions, k. That is, we evaluate the model at varying degrees of test-time interventions on the validation set, and then set this as the number of features used by the model. In Table 3, we present the results of this approach. In the table, we compare F-Act, which uses post-training hyperparameter tuning of k, to F-Act "selected only", which only uses only the selected features. We find that in most datasets, this enables an improvement over the "selected only" variant of the model. Exceptions include the finance and mice protein datasets, where F-Act "selected only" outperforms F-Act $\dot{0}.005$, which is within the standard deviation of our model's F1 score for those datasets. However, on other datasets, such as PBMC, the improvement of F-Act over F-Act "selected only" is rather notably greater than twice the standard deviation. Overall we find that the difference in average F1 score falls within the standard deviation of the model, indicating that the potential gains for this mechanism are limited, however the substantial gains on the PBMC dataset, as well as the small computational costs associated with its implementation, indicate that the mechanism is worth exploring when deploying F-Act.