

---

# Feature Shift Localization Network

---

Míriam Barrabés<sup>\* 1 2</sup> Daniel Mas Montserrat<sup>\* 1</sup> Kapal Dev<sup>2</sup> Alexander G. Ioannidis<sup>1 3</sup>

## Abstract

Feature shifts between data sources are present in many applications involving healthcare, biomedical, socioeconomic, financial, survey, and multi-sensor data, among others, where unharmonized heterogeneous data sources, noisy data measurements, or inconsistent processing and standardization pipelines can lead to erroneous features. Localizing shifted features is important to address the underlying cause of the shift and correct or filter the data to avoid degrading downstream analysis. While many techniques can detect distribution shifts, localizing the features originating them is still challenging, with current solutions being either inaccurate or not scalable to large and high-dimensional datasets. In this work, we introduce the Feature Shift Localization Network (FSL-Net), a neural network that can localize feature shifts in large and high-dimensional datasets in a fast and accurate manner. The network, trained with a large number of datasets, learns to extract the statistical properties of the datasets and can localize feature shifts from previously unseen datasets and shifts without the need for re-training. The code and ready-to-use trained model are available at <https://github.com/AI-sandbox/FSL-Net>.

## 1. Introduction

Feature distribution shifts between data sources are common in many real-world applications using multi-dimensional data composed of a set of “corrupted” features (i.e., dimensions) with mismatching statistical qualities between sources. These feature shifts are prevalent in healthcare,

biomedical, and life sciences datasets, where different samples are generated at different organizations (e.g., hospitals, labs), with differing lab technologies, hardware, and data processing producing unique structural biases. In clinical genomics, feature shifts can arise from heterogeneous data acquisition protocols, which may involve differences in genotyping arrays or phenotype curation procedures (Moreno-Grau et al., 2024). Similar shifts are found in social sciences, streaming, and e-business applications, where combining tabular and structured data from multiple sources, regions, and times without proper homogenization steps can lead to mismatching and biased features due to incorrect data collection procedures, human entry errors, faulty standardization, or erroneous data processing (Barchard & Pace, 2011; Dai et al., 2015). Similarly, multi-sensor applications in the manufacturing industry, medicinal monitoring, finance analysis, and defense can suffer feature shifts due to faulty sensors and measuring devices (Qian et al., 2022; Barrabés et al., 2023). While numerous techniques enable pre-processing data sources to reduce feature shifts, proper data homogenization can be a challenging task requiring data-dependent and domain-specific techniques (Lim et al., 2018). When data homogenization fails, unattended feature shifts can negatively impact downstream applications, leading to erroneous scientific results or biased machine learning models, which makes feature shift localization critical in many data-driven domains.

*Feature shift localization* is the task of enumerating which features of multi-dimensional datasets are originating the distribution shift between two or more data sources. The localization step is necessary to identify and correct the error source, whether by data removal or correction in tabular data-based applications or through physical intervention in multi-sensory scenario applications. Extensive literature on anomaly detection and distribution shift detection (Yu et al., 2018; Pan et al., 2020) has led to numerous tools for automated shift detection that are common in data-centric AI (DCAI) and machine learning systems (MLSys) technologies, providing functionalities for data quality control, homogenization, and monitoring (Ginart et al., 2022; Piano et al., 2022; Zha et al., 2023; Subasri et al., 2023). While many methods focus on asserting whether two datasets follow the same distribution, most do not localize the exact features causing the shift, and recent promising shift localization techniques still fail to scale to large and high-

---

<sup>\*</sup>Equal contribution <sup>1</sup>Department of Biomedical Data Science, Stanford University, Stanford, CA 94305 USA <sup>2</sup>Department of Computer Science, Munster Technological University, Cork T12 P928, Ireland <sup>3</sup>Department of Biomolecular Engineering, University of California Santa Cruz, Santa Cruz, CA 95060 USA. Correspondence to: Alexander G. Ioannidis <ioannidis@stanford.edu>.

dimensional datasets common in many areas (Kulinski et al., 2020; Barrabés et al., 2023).

In this work, we introduce a novel neural network, the *Feature Shift Localization Network* (FSL-Net), which can localize shifts with high accuracy while scaling to high-dimensional and large datasets. The network extracts statistical descriptors from two datasets and then processes them to localize the features originating the distribution shift. Namely, FSL-Net has two subnetworks, a *Statistical Descriptor Network* that compresses datasets into statistical functionals that summarize their underlying distribution, and a *Prediction Network* that combines the statistical descriptors between datasets to predict the probability of being *corrupted* for each feature. FSL-Net makes use of convolutional and pooling layers to achieve *invariance* to sample order and approximate *equivariance* to feature order. The network is trained end-to-end using multiple datasets with different types of simulated feature shifts and is evaluated on previously unseen datasets and shift types, showing that it generalizes well out-of-the-box to a large variety of data and shifts without the need for re-training. FSL-Net surpasses previous feature shift localization methods in localization accuracy and speed.

Our contributions include: (1) we propose a novel neural network architecture that provides *invariance* to sample order and *equivariance* to feature order while scaling to large and high-dimensional datasets; (2) we design a training approach leading to a network that generalizes to unseen data and shifts without the need for re-training; (3) we provide an in-depth experimental evaluation with multiple manipulation types, datasets, and network configurations.

## 2. Related Work

**Distribution Shift Detection.** The detection of distribution shifts consists of predicting if  $p \neq q$ , where  $p$  and  $q$  are the reference and query distributions, respectively. Numerous techniques exist for detecting distribution shifts in univariate distributions (Gama et al., 2014; Lu et al., 2018; Pan et al., 2020), and there is a growing focus on multivariate data (Rabanser et al., 2019), which can exhibit various types of shifts such as marginal, concept, covariate, or label shifts (Lu et al., 2016; Losing et al., 2016; Liu et al., 2020). Recent shift detection techniques include (Yu et al., 2018), which makes use of hypothesis testing for concept drift detection, and (Rabanser et al., 2019), which applies two-sample multivariate hypothesis testing via Maximum-Mean Discrepancy (MMD) (Gretton et al., 2012), univariate hypothesis tests with marginal Kolmogorov-Smirnov (KS) tests, and dimensionality reduction techniques.

**Feature Shift Localization.** While distribution shift detection techniques focus on detecting whether a shift exists

between distributions, feature shift localization methods aim to predict which features are causing the shift. A notable contribution in this area is the work by (Kulinski et al., 2020), which introduces a conditional test capable of accurately identifying shifted features with model-free and model-based approaches: K-Nearest Neighbors with KS statistic (KNN-KS), multivariate Gaussian with KS (MB-KS), multivariate Gaussian and Fisher-divergence test statistics (MB-SM), and deep density neural models with Fisher-divergence test (Deep-SM). DataFix (Barrabés et al., 2023) is a more recent method that improves localization accuracy by iteratively training a random forest to distinguish between reference and query distributions, removing the features with the highest impurity-based importance scores until divergence is minimized, and applying a knee-detection algorithm to determine the optimal stopping point. Although DataFix performs well in many cases, it struggles with detecting challenging feature shifts and scales poorly with high-dimensional and large datasets. Its repeated use of random forest training results in significant computational overhead, limiting its applicability to real-world applications involving massive datasets. In this paper, we adopt the same evaluation benchmark as DataFix and introduce FSL-Net to overcome these limitations.

**Feature Selection.** Feature selection methods localize the most relevant features for classification or regression, providing interpretability and removing redundancy. Wrapper (Maldonado & Weber, 2009; Mustaqeem et al., 2017), filtering (Nasir et al., 2020; Hopf & Reifenrath, 2021), and embedded methods (Tran et al., 2016; Huang et al., 2018) are among the most common techniques. Wrapper methods select features of interest by training ML models and adding or removing features through a search process. Filtering methods include Mutual Information (MI) (Battiti, 1994), ANOVA-F test (Elssied et al., 2014), Chi-square test (Bahassine et al., 2020), Minimum Redundancy Maximum Relevance (MRMR) (Ding & Peng, 2005; Li et al., 2018), and Fast-Conditional Mutual Information Maximization (FAST-CMIM) (Fleuret, 2004), among others. Such methods extract statistical information from the data to rank the importance of each feature. Embedded methods rank features using built-in scores from ML models, such as logistic regression weights (Cheng et al., 2006) or the mean decrease in impurity (Gini index) in random forests (Sylvester et al., 2018), selecting those with the highest scores.

**Optimal Transport.** Optimal transport (OT) theory compares probability distributions by computing the minimal cost required to transform one into another, inducing a meaningful distance that reflects both global structure and the geometry of the underlying space. Its formulation as a linear programming problem (Kantorovitch, 1958) connected OT to the broader field of optimization (Quanrud, 2018). Its relevance has since expanded across fields, including

computer vision (Izquierdo & Civera, 2024), economics (Galichon, 2018), logistics (Nadal-Roig & Plà-Aragónés, 2015), and statistical inference (Goldfeld et al., 2024). Recent advances in scalable numerical solvers, such as entropic regularization and Sinkhorn iterations, have enabled OT to scale to high-dimensional settings and find applications in data science (Peyré et al., 2019; Montesuma et al., 2024), generative modeling (Sanjabi et al., 2018), and domain adaptation (Courty et al., 2016). However, while OT techniques can characterize divergences, they do not provide a direct methodology to localize divergent features, which is the main focus of this paper, and would require modifications in order to be applied for the feature shift localization task.

**Data-centric AI.** Data-centric AI (DCAI) is the paradigm that encapsulates tools and techniques aimed at improving data quality and quantity to build robust, accurate, and efficient AI systems. Unlike model-centric approaches, which prioritize refining models while working with a fixed dataset, DCAI emphasizes improving datasets through systematic and iterative processes. Key aspects include expanding datasets through data collection (Ghosh & Kaviraj, 2023), annotation (Boecking et al., 2020; Caporali et al., 2023), augmentation (Montserrat et al., 2017; Geleta et al., 2023), and integration (Xiaojuan & Yu, 2023), as well as refining data through cleaning (Krishnan & Wu, 2019; Costanzo, 2023; Barrabés et al., 2024) and feature engineering (Sinaci et al., 2023; Buckley et al., 2023). The increasing complexity and scale of datasets have made automated methods indispensable for data refinement. A growing focus within DCAI is the localization of feature shifts (Zha et al., 2023; Barrabés et al., 2023).

**Deep Sets, Equivariant, and Invariant Networks.** Invariance and equivariance properties are important in many applications and have proved successful in modeling physics and chemical systems, with numerous neural networks developed to have such properties (Benton et al., 2020; Batzner et al., 2022; Ruhe et al., 2024). Graph neural networks and attention-based networks have been shown to provide similar properties (Lim & Nelson, 2022). Deep Sets (Zaheer et al., 2017) introduced an architecture for modeling set-structured data, ensuring invariance on the order of samples. Our proposed FSL-Net adopts similar design choices as Deep Sets to obtain sample-order invariance and approximate feature-order equivariance.

Section A provides a more detailed description of the benchmarking methods evaluated in this paper, including DataFix, MB-SM, MB-KS, KNN-KS, and Deep-SM, as well as MI, SelectKBest, MRMR, and Fast-CMIM.

### 3. Feature Shift Localization Network

**Problem Formulation.** We follow a similar problem formulation as described in (Kulinski et al., 2020; Barrabés et al., 2023), with minor adaptations:

**Definition 1. [Empirical Feature Shift Localization Task]** We are given two sets of  $d$ -dimensional samples  $X = \{x_1, x_2, \dots, x_N\}$  and  $Y = \{y_1, y_2, \dots, y_M\}$  from distributions  $p$  and  $q$  respectively, with  $x_i, y_i \in \mathbb{R}^d$ ,  $|X| = N$ , and  $|Y| = M$ . The feature shift localization task consists of predicting the subset of shifted features  $C$  from the input data  $X$  and  $Y$  using a mapping  $F$ , such that  $C = F(X, Y)$ , that satisfies  $D(p_{\bar{C}}, q_{\bar{C}}) = 0$ ,  $D(p, q) > 0$ , and  $C = \arg \min_{D(p_{\bar{C}}, q_{\bar{C}})=0} |C|$ , where  $D$  is a valid statistical distance.

A feature shift between  $d$ -dimensional distributions  $p$  and  $q$  is present if after removing the “corrupted” dimensions  $C$  and keeping only the “non-corrupted” dimensions  $S = \bar{C}$ , with  $d = |S| + |C|$ , the divergence between the restricted distributions is  $D(p_S, q_S) = 0$ . The number of corrupted features  $|C|$  is assumed to be unknown. We refer to  $X$  and  $Y$  as the “reference” and “query” datasets, and to  $p$  and  $q$  as the “reference” and “query” distributions, respectively. In practice,  $p$  and  $q$  are unknown and only accessible through the samples  $X$  and  $Y$ , requiring the task to be approximated by a method  $\tilde{F}$  that maps the input data to the predicted set of corrupted features:  $\hat{C} = \tilde{F}(X, Y)$ . Such method  $\tilde{F}$  can have the form of machine learning-based hypothesis testing (Kulinski et al., 2020), iterative heuristic algorithms as in DataFix (Barrabés et al., 2023), or the end-to-end trained parametric neural network introduced in this paper. Typically,  $\tilde{F}$  is designed to predict a set  $\hat{C}$  that is as close as possible to the true set of corrupted features  $C$ . This set can be represented either as a collection of positional indices or as a  $d$ -dimensional Boolean vector  $C \in \{0, 1\}^d$ , where each entry indicates whether a corresponding feature is corrupted or not. Note that the vector representation is equivariant to the feature ordering. In this paper, we use the set and vector notations interchangeably unless unclear from the context. We focus on scenarios where either no features or only a subset exhibit a shift. If all features experienced a shift, it would be impossible to determine whether the differences arise naturally between the reference and the query. Thus, we base our approach on the assumption that the true, unmodified query originates from the same distribution as the reference.

As described in (Barrabés et al., 2023), the presented definition of feature shifts covers a wide range of distribution shifts: marginal shifts with  $D(p_i, q_i) > 0$ , where  $p_i$  and  $q_i$  represent the marginal distribution of the  $i$ th dimension; correlation shifts with  $D(p, q) > 0$  and  $D(p_i, q_i) = 0$  for all  $i$ , where marginal distributions match but the multi-variate distributions do not; and similarly, correlation shifts with

$D(p_S, q_S) = 0$  and  $D(p_C, q_C) = 0$  but  $D(p, q) > 0$ , where correlations are maintained locally, but a shift is present when considering  $C$  and  $S$  simultaneously. Note that this framework can also model label shifts in regression or classification tasks by simply considering the label as an additional dimension of  $p$  and  $q$ .

The Feature Shift Localization Network has two main sub-networks: the Statistical Descriptor Network  $\phi_\theta$ , which gen-

## Statistical Functionals and Statistical Functional Maps.

4



the form  $\mu = \mathcal{A}(p) = \int g(x)dP(x)$ , which can be expressed as a simple average for empirical distributions:  $\mu_N = \mathcal{A}(p) = \frac{1}{N} \sum_{j=1}^N g(x_j)$ . Note that  $g(\cdot)$  does not need to be a linear mapping and can be any (potentially non-linear) function, even a neural network. Empirical mean and histogram estimates are examples of linear functionals.

In this paper, we extend the concept of statistical functionals to *statistical functional maps* – mappings  $T$  that project a  $d$ -dimensional multivariate CDF  $P$  (or equivalently a pdf  $p$ ) into a  $d \times t$  tensor  $\mu \in \mathbb{R}^{d \times t}$ , where the  $i$ th and  $k$ th component  $\mu_{i,k} = T^k(P, i)$  is obtained by applying the mapping  $T^k$  to the multivariate distribution  $P$  while using the positional information  $i$  (i.e., dimension index). An example of a statistical functional map is a tensor representing a histogram of  $t$  bins for each of the  $d$  marginal distributions, which can be computed as  $\mu_{i,k}^H = \frac{1}{N} \sum_{j=1}^N \mathbf{1}_{b_k}(x_{j,i})$ , with  $1 \leq k \leq t$  and  $1 \leq i \leq d$ , where  $x_{j,i}$  is the  $i$ th dimension of the  $j$ th sample, and  $\mathbf{1}_{b_k}(x)$  equals 1 if  $x$  is in the interval  $b_k$  defining the  $k$ th bin, and 0 otherwise. Other examples include tensors capturing the first  $t$  moments of the  $d$  marginal distributions or the  $d \times d$  covariance matrix  $\mu_{i,k}^C = \text{Cov}_p(i, k)$  of distribution  $p$ , with  $t = d$ . Statistical functional maps, indexed by dimension  $i$ , provide finite-dimensional summaries of multivariate distributions, where each  $i$ th component captures the statistical properties of the  $i$ th dimension of the distribution and its interactions with other dimensions.

**Statistical Descriptor Network.** The Statistical Descriptor Network is the first component of FSL-Net. This network extracts multiple statistical functional maps from the reference and query datasets, which are then fed into the Prediction Network to localize potential shifts. The Statistical Descriptor Network extracts three statistical functional maps: (1) a non-parametric map  $\mu_S = \phi_S(p_N)$ , with  $\mu_S \in \mathbb{R}^{d \times t_1}$ , consisting of simple statistical measures such as marginal means and histograms; (2) a map predicted by a parametric shallow network named Moment Extraction Network  $\mu_\tau = \phi_\tau(p_N)$ , with  $\mu_\tau \in \mathbb{R}^{d \times t_2}$ , designed to extract second and higher moments of the data; and (3) a final map extracted by a parametric deep residual network named Neural Embedding Network  $\mu_\omega = \phi_\omega(p_N)$ , with  $\mu_\omega \in \mathbb{R}^{d \times t_3}$ , designed to extract a richer representation of the dataset. The maps are concatenated, generating a unique map  $\mu_p \in \mathbb{R}^{d \times t}$  describing distribution  $p$  (and similarly for  $q$ ), with  $t = t_1 + t_2 + t_3$ . Note that  $d$  varies with each input dataset, but  $t$  remains fixed across datasets and only depends on the network hyperparameters. We use  $\mu_p = \phi_\theta(p_N) = [\mu_S; \mu_\tau; \mu_\omega]$  to denote the complete mapping of the three statistical functional maps for distribution  $p_N$ .

**Statistical Measures.** The first statistical functional map  $\mu_S = \phi_S(p_N)$  contains various measures that capture key statistical properties of individual features in the reference

and query datasets. Namely, these measures include the mean (indicating the average value), standard deviation (indicating the spread), median (indicating the midpoint), mean absolute deviation (indicating the average distance from the mean),  $p$ -order moments (capturing univariate higher-order characteristics), marginal histograms (approximating marginal pdfs), and empirical marginal CDFs. Table 1 presents the formulas for each measure. Note that except for histograms, empirical CDFs, and  $p$ -order moments, all measures can be represented with arrays of dimensionality  $d \times 1$ , while the histograms and empirical CDFs are represented in tensors of size  $d \times t_h$ , where  $t_h$  is the number of bins or powers  $p$  used. By concatenating all the measures, we obtain a statistical functional map  $\mu_S \in \mathbb{R}^{d \times t_1}$ .

Table 1. Statistical measures of the Statistical Descriptor Network.  $N$  is the number of observations in  $X$ ,  $x_j$  denotes the  $i$ th dimension of the  $j$ th sample (subscript  $i$  is omitted for brevity),  $\epsilon$  is a small positive constant for numerical stability,  $b_k$  and  $c_k$  are histogram intervals and CDF thresholds, respectively, and  $\bar{x}$  and  $\sigma$  are the empirical mean and standard deviation of the  $i$ th dimension.

Statistical Measure	Linear	Equation $\mu_{i,k}$
Mean	Yes	$\bar{x} = \frac{1}{N} \sum_{j=1}^N x_j$
Standard Deviation	No	$\sigma = \sqrt{\frac{1}{N} \sum_{j=1}^N (x_j - \bar{x})^2}$
Median	No	$x_{(\frac{N}{2})}$
Mean Absolute Deviation	Yes	$\frac{1}{N} \sum_{j=1}^N  x_j - \bar{x} $
$p$ -order Moments	Yes	$\frac{1}{N} \sum_{j=1}^N x_j^p$
Histogram	Yes	$\frac{1}{N} \sum_{j=1}^N \mathbf{1}(x_j \in b_k)$
Empirical CDF	Yes	$\frac{1}{N} \sum_{j=1}^N \mathbf{1}(x_j \leq c_k)$

**Moment Extraction Network.** The previously described statistical measures work well for capturing marginal distributions, but they fail to capture correlations and higher-order relations between dimensions. While the covariance matrix could address this, it becomes impractical for high-dimensional datasets due to its quadratic growth with the number of features. In order to capture higher-order relations between features, we make use of the Moment Extraction Network  $\mu_\tau = \phi_\tau(p_N) = \frac{1}{N} \sum_{i=1}^N (Wx_i + b)_+$ , where  $(W, b)$  represent an affine mapping, and  $(\cdot)_+$  is the ReLU activation function. In order to adapt to changing input dimensionalities  $d$ , the affine mapping is parametrized using a convolutional layer and a batch normalization layer. By applying padding, the output of the network has dimensionality  $d \times t_2$ , where  $t_2$  is the number of output channels of the convolution. The convolutional and ReLU layers are applied to each sample  $x_j$  independently, and a sample-wise mean pooling operation is applied to obtain a dataset-level vector. Note that this network can be seen as approximating generalized moments of the data (Gretton et al., 2012; Li et al., 2015; Perera et al., 2022). Both the Moment Extraction Network and the Neural Embedding Network are trained jointly with the Prediction Network by using a cross-entropy loss and auxiliary loss (see the *loss functions* Section).

**Neural Embedding Network.** To complement the statistical measures and the Moment Extraction Network, we include a convolutional deep residual network to predict a linear statistical functional map:  $\mu_\omega = \phi_\omega(p_N) = \frac{1}{N} \sum_{j=1}^N \omega_\theta(x_j)$ .

The Neural Embedding Network, as well as the Prediction Network detailed below, is built upon residual blocks. Each residual block consists of alternating 1D-convolutional layers across features, batch normalization (BN), and a hyperbolic tangent activation function (Tanh). Skip connections are incorporated to facilitate efficient information flow and mitigate vanishing gradients. We conducted experiments with attention-based Multi-Layer Perceptrons (MLPs) using unit-kernel convolutions, but a fully convolutional design worked best (see Section D).

The  $i$ th output of the statistical functional map predicted by the Neural Embedding Network has the form  $\mu_{\omega,i} = \frac{1}{N} \sum_{j=1}^N \left( W x_{j,i} + \sum_{h=1}^H \omega_h(x_j) \right)$ , where  $W x_{j,i}$  is an affine transformation of the  $i$ th input feature, and  $\omega_h(x_j)$  denotes the output of each of the  $H$  residual blocks, capturing non-linear relationships between features.

The complete Statistical Descriptor Network is applied to  $X$  and  $Y$ , yielding  $\mu_p = \phi_\theta(p_N) = \phi_\theta(X)$  and  $\mu_q = \phi_\theta(q_M) = \phi_\theta(Y)$ . Note that  $\phi_\theta(\cdot)$  is shared between the reference and the query. These descriptors are then fed into the Prediction Network.

**Prediction Network.** The Prediction Network  $\gamma_\theta$  combines the statistical functional maps  $\mu_p = \phi_\theta(p_N)$  and  $\mu_q = \phi_\theta(q_M)$  to predict a vector of probabilities  $\hat{P} = \gamma_\theta(\mu_p, \mu_q)$ , indicating the likelihood of each feature belonging to the corrupted set  $C$ . These maps are combined through an operation  $\alpha(\cdot)$ , producing a joint statistical map  $\mu_{p,q} = \alpha(\mu_p, \mu_q)$ . After evaluating various merging operations (see Section E.3), we selected the normalized squared difference:  $\alpha(\mu_p, \mu_q) = \frac{(\mu_p - \mu_q)^2}{\|\mu_p\| + \epsilon}$ , where  $\epsilon$  is a small positive constant for numerical stability. This approach enables the network to compare statistical maps in a manner that accounts for their relative magnitudes. The resulting joint representation has dimensions  $\mu_{p,q} \in \mathbb{R}^{d \times t}$ . The Prediction Network then applies multiple residual blocks to the joint map  $\mu_{p,q}$  and produces the final probability estimates through a sigmoid activation layer. Note that, by employing the squared difference, the Prediction Network observes a difference between statistical maps resembling the Maximum Mean Discrepancy (MMD) metric (Gretton et al., 2012; Li et al., 2015) and acts as a mapping from a distance between distributions into shift probability estimates.

The complete structure of the feature shift localization network has the following form:

$$\hat{P} = \psi_\theta(p_N, q_M) = \psi_\theta(X, Y) = \gamma_\theta(\phi_\theta(p_N), \phi_\theta(q_M)) \quad (1)$$

**Loss Functions.** FSL-Net is trained end-to-end using the binary cross-entropy loss between the predicted probabilities  $\hat{P}$  and the ground truth corrupted feature set  $C$ :  $\ell_{CE}(C, \hat{P}) = \sum_{k=1}^d C_k \log(\hat{P}_k) + (1 - C_k) \log(1 - \hat{P}_k)$ . We also add an auxiliary loss function to the predicted statistical functional maps to encourage the learning of useful discriminative features and enforce locality:  $\ell_{aux}(C, \mu_p, \mu_q) = \frac{\|\mu_p \bar{C} - \mu_q \bar{C}\|^2}{\|\mu_p \bar{C} - \mu_q \bar{C}\|^2}$ . The loss is computed and averaged across multiple labeled datasets with simulated shifts, denoted as  $D = \{(C^{(z)}, X^{(z)}, Y^{(z)}) : 1 \leq z \leq N_D\}$ , resulting in the total loss function:  $\mathcal{L}(\psi_\theta, D) = \sum_{z=1}^{N_d} \ell_{CE}(C^{(z)}, \psi_\theta(X^{(z)}, Y^{(z)})) + \lambda \ell_{aux}(C^{(z)}, \phi_\theta(X^{(z)}), \phi_\theta(Y^{(z)}))$ . In practice,  $\mathcal{L}(\psi_\theta, D)$  is approximated by using mini-batches and gradient accumulation and optimized with the Adam optimizer.

**Sample-wise Invariance, Feature-wise Equivariance, and Locality.** Ensuring feature equivariance and sample invariance can help neural networks generalize across datasets with varying numbers of features and samples; (a) Sample-wise invariance: The mean pooling operation used in the linear functionals, Neural Embedding, and Moment Extraction Networks provides maps with shapes that are independent of dataset size, allowing FSL-Net to generalize across datasets of different dimensions. Additionally, the non-linear statistical functionals are computed with the samples sorted from smallest to largest, making them invariant to sample order; (b) Feature-wise invariance: The statistical measures are applied to the marginal distributions, making them equivariant to feature ordering. Convolutions, however, are not typically feature-wise equivariant. Instead, FSL-Net approximates feature invariance by shuffling features in each training mini-batch, enforcing learned representations to be invariant to feature order; (c) Locality: We enforce that the  $i$ th dimension of the statistical functional maps primarily captures the statistical properties of the  $i$ th input feature by (1) using marginal distribution-based statistical measures, (2) incorporating a residual connection in the Neural Embedding Network, and (3) applying the auxiliary loss to the statistical functional maps.

**Training and Validation Datasets.** We source a total of 1,032 diverse tabular datasets from OpenML (Van Rijn et al., 2013), with 10–28k features and 500–3.6M samples, covering continuous, categorical, and mixed data types. Additionally, we generate 368 simulated datasets: 184 based on probabilistic distributions (Gaussian, Bernoulli, and Beta mixtures) and 184 from algebraic functions (Polynomial, Sine, and Logarithmic), each with 5,000 samples and 1,000 features. In total, 1,350 datasets are used for training, with

Table 2. Manipulation types applied to continuous and/or categorical features during training and validation.

Type	Mapping	Description	Shift	Data
T1	$\beta_i x_i$	Each value is multiplied by a random number between 0 and 1.	$p_i \neq q_i$	Cont.
T2	$\beta_i(1-x) + (1-\beta_i)x$	Each value is replaced by a linear combination of $x$ and its negation.	$p_i \neq q_i$	Cont.
T3	$\beta_i \sim \text{Uniform}(0, 1)$ $x_i \sim p_{N_i}$	The $i$ th feature of $Y$ is replaced by sampling from the ref. empirical marginal distribution $p_{N_i}$ .	$p_i = q_i, p_C \neq q_C$ , $q_C = \prod_{i \in C} q_i$	Both
T4	$\text{clamp}_{0,1}(x + \epsilon)$	Add Gaussian noise with $\mu \sim \text{Uni}(-0.2, 0.2)$ and $\sigma \sim \text{Uni}(0.001, 0.5)$ .	$p_i \neq q_i, \mathbb{E}[q_i] \approx \mathbb{E}[p_i] + \mu$	Cont.
T5	$\epsilon \sim \text{Normal}(\mu, \sigma)$ $\text{CNN}(x)$	Forward through a CNN with min-max normalization or binarization.	$p_i \neq q_i$	Both
T6	$x_C \sim p_{N_C}$	Similar to manipulation (c), but all the features within $C$ are sampled simultaneously from $p_{N_C}$ .	$p_i = q_i, p_C = q_C, p \neq q$	Both
T7	$\text{KNN}(x)$ $K = \{1-4, 7-9\}$	Predict feature with KNN (Regressor).	-	Cont.
T8	$\text{KNN}(x)$ $K = \{1-4, 7-9\}$	Predict feature with KNN (Classifier).	-	Cat.

50 reserved for validation. Section B describes in detail the dataset selection, preprocessing, and simulation procedures.

### Training and Validation Manipulations Simulations.

During training, datasets are shuffled in sample and feature order, and random subsets of samples (from 1,000 to 10,000) and features (from 8 to 256) are selected, with each feature normalized to a range of 0 to 1. Each subset is then split equally into reference and query samples. In the query set, a random subset of features (up to 25%) is manipulated based on feature type (continuous or categorical). Validation batches follow the same process but are limited to 2,048 features. Manipulations, outlined in Table 2 (T1 to T8), are selected with probabilities inversely proportional to the validation F-1 score performance observed during training for each given type. Multiple manipulations are used to simulate a wide range of feature shifts. Note that the manipulations applied during training and validation differ from those applied to the test set.

## 4. Experimental Results

**Evaluation Setup.** Our evaluation setup is consistent with (Barrabés et al., 2023), using the same reference and query sets and optimized benchmarking methods. Hyperparameter tuning for FSL-Net is detailed in Section E. We compare FSL-Net against five feature shift localization methods (DataFix, MB-SM, MB-KS, KNN-KS, and Deep-SM) and four feature selection methods (MI, SelectKBest, MRMR, and Fast-CMIM). The SelectKBest method employs the Chi-square test for categorical datasets and the ANOVA-F test for continuous datasets. For MB-SM, MB-KS, KNN-KS, and Deep-SM, the number of manipulated features  $|C|$  is provided, while the other methods, including FSL-Net, do not require it, reflecting a more realistic setting.

**Evaluation Data.** We use the same evaluation manipulations and evaluation datasets as in (Barrabés et al., 2023), with the Gas, Covid, and Energy datasets also aligning with

Table 3. Datasets used during benchmarking.

Dataset Type	Dataset	# Features	# Samples
Continuous	Gas	8	12,815
	Covid	10	9,889
	Energy	26	19,735
	Musk2	166	6,598
	Scene	294	2,407
	MNIST	784	70,000
	Polynomial	1,000	10,000
	Cosine	1,000	10,000
Categorical	Dilbert	2,000	10,000
	Phenotypes	1,227	31,424
	Founders	10,000	4,144
	Canine	198,473	1,444

those in (Kulinski et al., 2020). The benchmark datasets vary in data type, feature dimensionality (from 8 to 198,473), and sample size (from 1,444 to 70,000). Table 3 provides an overview of each dataset, detailing its data type (continuous or categorical), sample size, and feature dimensionality. The continuous datasets are sourced from the UCI repository (Gas (Huerta et al., 2016), Energy (Candanedo et al., 2017), and Musk2 (Blake, 1998)) and OpenML (Scene (Boutell et al., 2004), MNIST (Deng, 2012), and Dilbert (Vanschoren et al., 2014)). Additionally, a Covid-19 dataset (Force, 2022) and two simulated datasets generated from algebraic functions that differ from those used in our training and validation simulations (Cosine and Polynomial) (Barrabés et al., 2023) are included. The categorical datasets consist of high-dimensional biomedical data, including the Phenotypes dataset (Qian et al., 2020), a subset of categorical traits from the UK Biobank, the Founders dataset containing binary-coded human DNA sequences (Perera et al., 2022), and the Canine dataset comprising binary-coded dog DNA sequences (Barrabés et al., 2023). Each dataset is normalized on a per-feature basis to a range of 0 to 1, and the samples are evenly divided into two subsets, forming the reference and query sets. As in (Barrabés et al., 2023), a random fraction of query features (5%, 10%, or 25%) undergoes one of 10 manipulation types for continuous data or

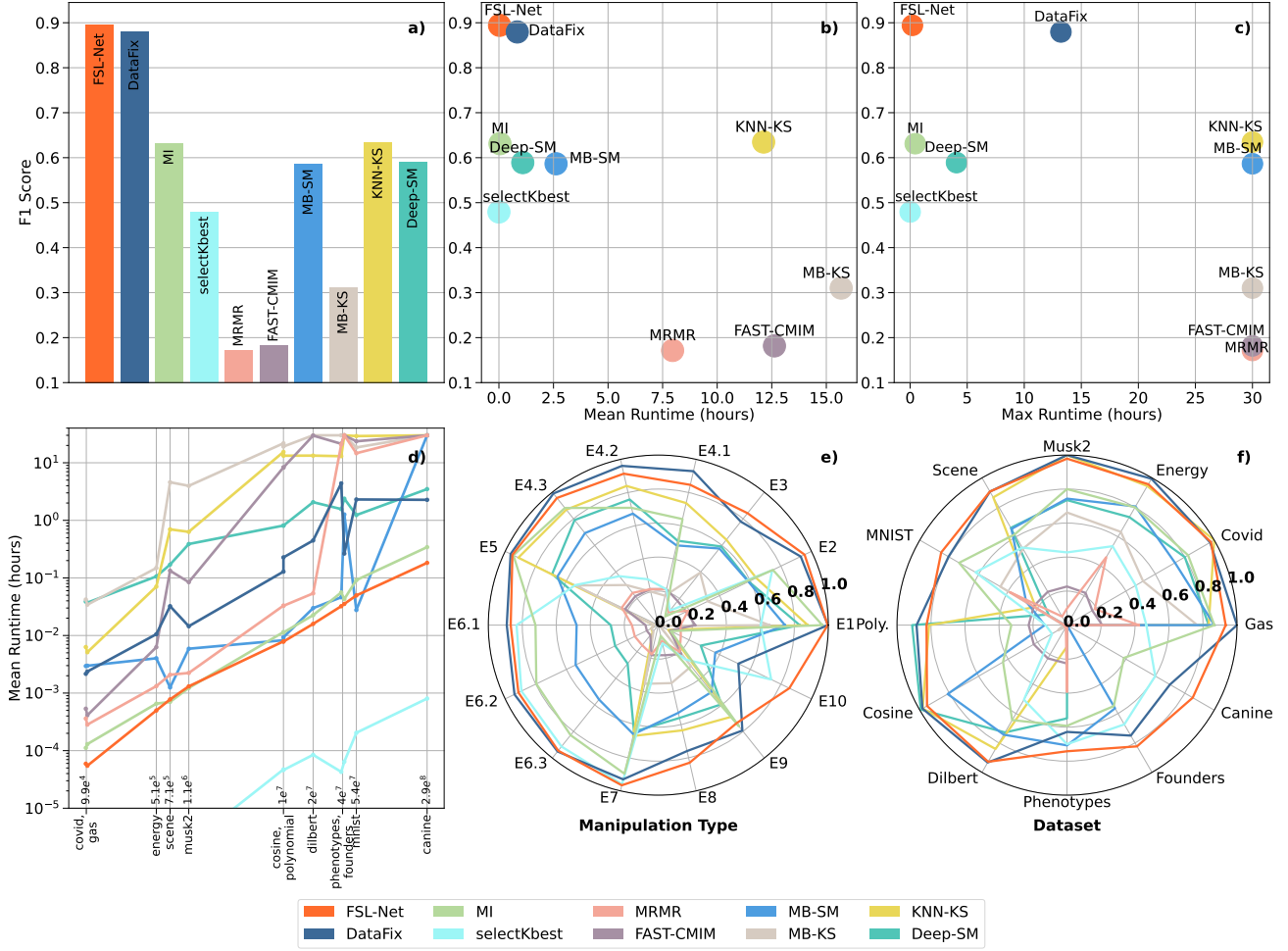


Figure 2. Performance and runtime comparison across feature shift localization methods: a) mean F-1 scores across manipulation types, fractions of manipulated features, and datasets; b) mean F-1 scores vs. mean runtime; c) mean F-1 scores vs. maximum runtime; d) mean runtime vs. sample-feature size product per dataset; e) mean F-1 scores by manipulation type; f) mean F-1 scores by dataset.

8 for categorical data (referred to as manipulations E1-E10). This process generates 30 query sets for continuous data and 24 for categorical data, each with a unique manipulation applied to a given feature subset. Section C details the evaluation manipulation types. Note that both testing datasets and manipulations differ from those used during FSL-Net training and validation (see Table 2).

**Evaluation Protocol and Hardware Specifications.** Performance is evaluated using the F-1 score for feature shift localization accuracy and wall-clock runtime for computational efficiency. Each experiment is run for up to 30 hours, which prevents some methods from completing evaluations on large datasets. To handle incomplete evaluations, we impute missing results with the lowest F-1 score from the same experiment among competing methods and assign them a 30-hour limit. All evaluations were conducted on an Intel Xeon Gold with 12 CPU cores.

**Feature Shift Localization Performance and Runtime Comparison.** Figure 2 presents the performance and runtime comparison across different methods. Figure 2a presents the average F-1 scores of feature shift localization computed across manipulation types, fractions of manipulated features, and datasets. Figure 2b depicts the mean F-1 scores against the mean runtime in hours, while Figure 2c illustrates the F-1 scores against the maximum runtime, highlighting the worst-case computational demands. Higher positions in these plots indicate better performance, whereas leftward positions indicate lower runtimes, emphasizing the balance between effectiveness and efficiency. SelectKbest and MI, despite their efficiency, exhibit poor performance in feature shift localization. Methods such as MRMR and FAST-CMIM struggle with scalability and display limited localization capabilities. Notably, FSL-Net achieves the highest average F-1 score, with feature shift localization performance comparable to that of DataFix but at significantly



faster speeds. Specifically, FSL-Net is approximately  $36\times$  faster on average than DataFix, and up to  $136\times$  on the high-dimensional Phenotypes dataset. Additionally, FSL-Net significantly outperforms MB-SM, MB-KS, KNN-KS, and Deep-SM, even though these methods have the advantage of accessing the ground truth  $|C|$ , while also demonstrating substantially faster computational speeds. While DataFix relies on a computationally intensive iterative optimization heuristic, and MB-SM, MB-KS, KNN-KS, and Deep-SM require dataset-specific training, FSL-Net only requires performing a forward pass through the network for each dataset, providing highly fast inference. Figure 2d shows the mean runtime as a function of the product between the sample and feature sizes for each dataset, confirming that FSL-Net consistently outperforms DataFix in speed across all datasets, ranking as the second fastest method after selectKBest, comparable to MI, but with much higher accuracy. Namely, DataFix requires up to 13 hours on the high-dimensional Phenotypes dataset from the UK Biobank and 9 hours on the Canine dataset, whereas FSL-Net completes these tasks in just 2 minutes and 12 minutes, respectively. These results highlight the practical advantage of FSL-Net: it is well-suited for processing high-dimensional large databases, such as those commonly found in biomedicine and e-commerce, making it a valuable alternative to DataFix due to its efficient scalability and excellent localization accuracy.

#### Performance across Manipulation Types and Datasets.

Figure 2e and Figure 2f show the mean F-1 scores categorized by type of manipulation and dataset, respectively. FSL-Net consistently matches or exceeds the performance of competing methods across all manipulation types, except for E9 and E4, where DataFix exhibits superior performance. These manipulations introduce only minimal perturbations that FSL-Net fails to localize, though lower probability thresholds may improve detection. DataFix fails to accurately detect E10, a shortcoming effectively addressed by FSL-Net. Methods relying on univariate tests, such as MRMR and Fast-CMIM, perform well for manipulations causing marginal distribution shifts but fail entirely with manipulations affecting feature correlations (manipulations E3 and E8). In contrast, techniques based on conditional testing (MB-SM, MB-KS, KNN-KS, and Deep-SM), along with DataFix and FSL-Net, successfully identify these more complex manipulations. FSL-Net shows a clear advantage in high-dimensional datasets (Phenotypes, Founders, and Canine), outperforming all competing methods and highlighting its effectiveness in handling large feature sets.

**Ablation Analysis.** We assess the impact of each component of FSL-Net’s Statistical Descriptor Network by training models with different combinations of its three components: Statistical Measures (SM), Moment Extraction Network (ME), and Neural Embedding Network (NE). All variants include the Prediction Network (PN), except for the SM-

only baseline, where predictions are obtained by directly thresholding the statistical measures. Table 4 presents the mean F-1 score achieved by each configuration. Training lasted three days for all variants, except for the full model, which was trained for an extended period of seven days. Using the Prediction Network in the SM-only baseline greatly improves feature shift localization performance compared to simply thresholding the statistical measures (from an F-1 score of 0.307 to 0.710), emphasizing the crucial importance of the Prediction Network. Each component contributes to performance improvements, suggesting that each component has unique value, with extended training resulting in additional gains.

Table 4. Mean F-1 scores for various configurations of FSL-Net.

Training Duration	Model Configuration	F-1 Score
3-Day	SM	0.307
	SM + PN	0.710
	ME + PN	0.742
	NE + PN	0.783
	ME + NE + PN	0.770
	SM + ME + PN	0.855
	SM + NE + PN	0.878
	SM + ME + NE + PN	0.889
7-Day	SM + ME + NE + PN	<b>0.894</b>

Additional experimental results are presented in Section F, including: (1) a median-based evaluation of feature shift localization performance and runtime across different methods; (2) a detailed performance and efficiency comparison between FSL-Net and DataFix; (3) extended evaluations of FSL-Net and DataFix on high-dimensional image datasets (CIFAR10 and COIL-100); (4) an analysis of FSL-Net’s runtime improvement over DataFix; (5) an evaluation of the threshold-based variant of the SM-only baseline; and (6) a qualitative evaluation of FSL-Net on the MNIST dataset.

## 5. Conclusions

Current feature shift localization methods involve a trade-off between speed and accuracy when dealing with large data volumes. In this work, we introduced FSL-Net, a novel equivariant neural network that matches or surpasses existing state-of-the-art methods while offering a significant reduction in processing times. FSL-Net leverages neural-learned statistical descriptors, augmented with traditional statistical measures, to effectively capture the input distributions. By contrasting these descriptors across different data sources, the network accurately detects both univariate and multivariate shifts. Most importantly, FSL-Net is designed to manage datasets of varying sizes and to generalize across new distributions without the need for model re-training with each new estimation, providing significant speed and scalability advantages over competing methods.

## Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

## References

- Bahassine, S., Madani, A., Al-Sarem, M., and Kissi, M. Feature selection using an improved chi-square for arabic text classification. *Journal of King Saud University-Computer and Information Sciences*, 32(2):225–231, 2020.
- Barchard, K. A. and Pace, L. A. Preventing human error: The impact of data entry methods on data accuracy and statistical results. *Computers in Human Behavior*, 27(5): 1834–1839, 2011.
- Barrabés, M., Bonet, D., Moriano, V. N., Giró-i Nieto, X., Montserrat, D. M., and Ioannidis, A. G. Genomic databases homogenization with machine learning. In *2023 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pp. 2952–2959. IEEE, 2023.
- Barrabés, M., Mas Montserrat, D., Geleta, M., Giró-i Nieto, X., and Ioannidis, A. Adversarial learning for feature shift detection and correction. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S. (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 57597–57638. Curran Associates, Inc., 2023.
- Barrabés, M., Perera, M., Moriano, V. N., Giró-I-Nieto, X., Montserrat, D. M., and Ioannidis, A. G. Advances in biomedical missing data imputation: A survey. *IEEE Access*, pp. 1–1, 2024. doi: 10.1109/ACCESS.2024.3516506.
- Battiti, R. Using mutual information for selecting features in supervised neural net learning. *IEEE Transactions on neural networks*, 5(4):537–550, 1994.
- Batzner, S., Musaelian, A., Sun, L., Geiger, M., Mailoa, J. P., Kornbluth, M., Molinari, N., Smidt, T. E., and Kozinsky, B. E (3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *Nature communications*, 13(1):2453, 2022.
- Benton, G., Finzi, M., Izmailov, P., and Wilson, A. G. Learning invariances in neural networks from training data. *Advances in neural information processing systems*, 33: 17605–17616, 2020.
- Blake, C. Uci repository of machine learning databases. <http://www.ics.uci.edu/~mlearn/MLRepository.html>, 1998.
- Boecking, B., Neiswanger, W., Xing, E., and Dubrawski, A. Interactive weak supervision: Learning useful heuristics for data labeling. *arXiv preprint arXiv:2012.06046*, 2020.
- Boutell, M. R., Luo, J., Shen, X., and Brown, C. M. Learning multi-label scene classification. *Pattern recognition*, 37(9):1757–1771, 2004.
- Buckley, T., Ghosh, B., and Pakrashi, V. A feature extraction & selection benchmark for structural health monitoring. *Structural Health Monitoring*, 22(3):2082–2127, 2023.
- Candanedo, L. M., Feldheim, V., and Deramaix, D. Data driven prediction models of energy use of appliances in a low-energy house. *Energy and buildings*, 140:81–97, 2017.
- Caporali, A., Pantano, M., Janisch, L., Regulin, D., Palli, G., and Lee, D. A weakly supervised semi-automatic image labeling approach for deformable linear objects. *IEEE Robotics and Automation Letters*, 8(2):1013–1020, 2023.
- Cheng, Q., Varshney, P. K., and Arora, M. K. Logistic regression for feature selection and soft classification of remote sensing data. *IEEE Geoscience and Remote Sensing Letters*, 3(4):491–494, 2006.
- Costanzo, L. Data cleaning during the research data management process. *Research Data Management in the Canadian Context*, 2023.
- Courty, N., Flamary, R., Tuia, D., and Rakotomamonjy, A. Optimal transport for domain adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 39(9): 1853–1865, 2016.
- Dai, T., Hu, H., Wan, Y., Chen, Q., and Wang, Y. A data quality management and control framework and model for health decision support. In *2015 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, pp. 1792–1796, 2015. doi: 10.1109/FSKD.2015.7382218.
- Deng, L. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.
- Ding, C. and Peng, H. Minimum redundancy feature selection from microarray gene expression data. *Journal of bioinformatics and computational biology*, 3(02):185–205, 2005.
- Elssied, N. O. F., Ibrahim, O., and Osman, A. H. A novel feature selection based on one-way anova f-test for e-mail spam classification. *Research Journal of Applied Sciences, Engineering and Technology*, 7(3):625–638, 2014.

- Fleuret, F. Fast binary feature selection with conditional mutual information. *Journal of Machine learning research*, 5(9), 2004.
- Force, C. C. T. United states covid-19 cases and deaths by state over time. 2022.
- Galichon, A. *Optimal transport methods in economics*. Princeton University Press, 2018.
- Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., and Bouchachia, A. A survey on concept drift adaptation. *ACM computing surveys (CSUR)*, 46(4):1–37, 2014.
- Geleta, M., Mas Montserrat, D., Giro-i Nieto, X., and Ioannidis, A. G. Deep variational autoencoders for population genetics. *bioRxiv*, pp. 2023–09, 2023.
- Ghosh, E. and Kaviraj, T. Data collection: Best practices and techniques of footprint collection for newborn identification. *International Journal of Creative Research and Thoughts (IJCRT)*, 11(9), 2023.
- Ginart, T., Zhang, M. J., and Zou, J. Mldemon: Deployment monitoring for machine learning systems. In *International Conference on Artificial Intelligence and Statistics*, pp. 3962–3997. PMLR, 2022.
- Goldfeld, Z., Kato, K., Rioux, G., and Sadhu, R. Statistical inference with regularized optimal transport. *Information and Inference: A Journal of the IMA*, 13(1):iaad056, 2024.
- Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012.
- Hopf, K. and Reifenrath, S. Filter methods for feature selection in supervised machine learning applications—review and benchmark. *arXiv preprint arXiv:2111.12140*, 2021.
- Huang, X., Zhang, L., Wang, B., Li, F., and Zhang, Z. Feature clustering based support vector machine recursive feature elimination for gene selection. *Applied Intelligence*, 48(3):594–607, 2018.
- Huerta, R., Mosqueiro, T., Fonollosa, J., Rulkov, N. F., and Rodriguez-Lujan, I. Online decorrelation of humidity and temperature in chemical sensors for continuous monitoring. *Chemometrics and Intelligent Laboratory Systems*, 157:169–176, 2016.
- Izquierdo, S. and Civera, J. Optimal transport aggregation for visual place recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 17658–17668, 2024.
- Kantorovitch, L. On the translocation of masses. *Management science*, 5(1):1–4, 1958.
- Krishnan, S. and Wu, E. A. Automatic generation of data cleaning pipelines. *arXiv preprint arXiv:1904.11827*, 2019.
- Kulinski, S., Bagchi, S., and Inouye, D. I. Feature shift detection: Localizing which features have shifted via conditional distribution tests. *Advances in Neural Information Processing Systems*, 33:19523–19533, 2020.
- Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., and Liu, H. Feature selection: A data perspective. *ACM Computing Surveys (CSUR)*, 50(6):94, 2018.
- Li, Y., Swersky, K., and Zemel, R. Generative moment matching networks. In *International conference on machine learning*, pp. 1718–1727. PMLR, 2015.
- Lim, L.-H. and Nelson, B. J. What is an equivariant neural network. *arXiv preprint arXiv:2205.07362*, pp. 5987–6001, 2022.
- Lim, S. B., Tan, S. J., Lim, W.-T., and Lim, C. T. A merged lung cancer transcriptome dataset for clinical predictive modeling. *Scientific data*, 5(1):1–8, 2018.
- Liu, A., Lu, J., and Zhang, G. Diverse instance-weighting ensemble based on region drift disagreement for concept drift adaptation. *IEEE transactions on neural networks and learning systems*, 32(1):293–307, 2020.
- Liu, L., Liu, J., and Han, J. Multi-head or single-head? an empirical comparison for transformer training. *arXiv preprint arXiv:2106.09650*, 2021.
- Losing, V., Hammer, B., and Wersing, H. Knn classifier with self adjusting memory for heterogeneous concept drift. In *2016 IEEE 16th international conference on data mining (ICDM)*, pp. 291–300. IEEE, 2016.
- Lu, J., Liu, A., Dong, F., Gu, F., Gama, J., and Zhang, G. Learning under concept drift: A review. *IEEE transactions on knowledge and data engineering*, 31(12):2346–2363, 2018.
- Lu, N., Lu, J., Zhang, G., and De Mantaras, R. L. A concept drift-tolerant case-base editing technique. *Artificial Intelligence*, 230:108–133, 2016.
- Maldonado, S. and Weber, R. A wrapper method for feature selection using support vector machines. *Information Sciences*, 179(13):2208–2217, 2009.
- Montesuma, E. F., Mboula, F. M. N., and Souloumiac, A. Recent advances in optimal transport for machine learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

- Montserrat, D. M., Lin, Q., Allebach, J., and Delp, E. J. Training object detection and recognition cnn models using data augmentation. *Electronic Imaging*, 2017(10): 27–36, 2017.
- Moreno-Grau, S., Vernekar, M., Lopez-Pineda, A., Mas-Montserrat, D., Barrabés, M., Quinto-Cortés, C. D., Moatamed, B., Lee, M. T. M., Yu, Z., Numakura, K., et al. Polygenic risk score portability for common diseases across genetically diverse populations. *Human Genomics*, 18(1):93, 2024.
- Mustaqeem, A., Anwar, S. M., Majid, M., and Khan, A. R. Wrapper method for feature selection to classify cardiac arrhythmia. In *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 3656–3659. IEEE, 2017.
- Nadal-Roig, E. and Plà-Aragónés, L. M. Optimal transport planning for the supply to a fruit logistic centre. *Handbook of operations research in agriculture and the agri-food industry*, pp. 163–177, 2015.
- Nasir, I. M., Khan, M. A., Yasmin, M., Shah, J. H., Gabryel, M., Scherer, R., and Damaševičius, R. Pearson correlation-based feature selection for document classification using balanced training. *Sensors*, 20(23):6793, 2020.
- Pan, J., Pham, V., Dorairaj, M., Chen, H., and Lee, J.-Y. Adversarial validation approach to concept drift problem in user targeting automation systems at uber. *arXiv preprint arXiv:2004.03045*, 2020.
- Perera, M., Montserrat, D. M., Barrabés, M., Geleta, M., Giró-i Nieto, X., and Ioannidis, A. G. Generative moment matching networks for genotype simulation. In *2022 44th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, pp. 1379–1383. IEEE, 2022.
- Peyré, G., Cuturi, M., et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.
- Piano, L., Garcea, F., Gatteschi, V., Lamberti, F., and Morra, L. Detecting drift in deep learning: A methodology primer. *IT Professional*, 24(5):53–60, 2022.
- Qian, J., Tanigawa, Y., Du, W., Aguirre, M., Chang, C., Tibshirani, R., Rivas, M. A., and Hastie, T. A fast and scalable framework for large-scale and ultrahigh-dimensional sparse regression with application to the uk biobank. *PLoS genetics*, 16(10):e1009141, 2020.
- Qian, J., Song, Z., Yao, Y., Zhu, Z., and Zhang, X. A review on autoencoder based representation learning for fault detection and diagnosis in industrial processes. *Chemo-metrics and Intelligent Laboratory Systems*, pp. 104711, 2022.
- Quanrud, K. Approximating optimal transport with linear programs. *arXiv preprint arXiv:1810.05957*, 2018.
- Rabanser, S., Günnemann, S., and Lipton, Z. Failing loudly: An empirical study of methods for detecting dataset shift. *Advances in Neural Information Processing Systems*, 32, 2019.
- Ruhe, D., Brandstetter, J., and Forré, P. Clifford group equivariant neural networks. *Advances in Neural Information Processing Systems*, 36, 2024.
- Sanjabi, M., Ba, J., Razaviyayn, M., and Lee, J. D. On the convergence and robustness of training gans with regularized optimal transport. *Advances in Neural Information Processing Systems*, 31, 2018.
- Sinaci, A. A., Gencturk, M., Teoman, H. A., Laleci Erturkmen, G. B., Alvarez-Romero, C., Martinez-Garcia, A., Poblador-Plou, B., Carmona-Pérez, J., Löbe, M., and Parra-Calderon, C. L. A data transformation methodology to create findable, accessible, interoperable, and reusable health data: software design, development, and evaluation study. *Journal of medical Internet research*, 25:e42822, 2023.
- Subasri, V., Krishnan, A., Dhalla, A., Pandya, D., Malkin, D., Razak, F., Verma, A., Goldenberg, A., and Dolatabadi, E. Diagnosing and remediating harmful data shifts for the responsible deployment of clinical ai models. *medRxiv*, pp. 2023–03, 2023.
- Sylvester, E. V., Bentzen, P., Bradbury, I. R., Clément, M., Pearce, J., Horne, J., and Beiko, R. G. Applications of random forest feature selection for fine-scale genetic population assignment. *Evolutionary applications*, 11(2): 153–165, 2018.
- Tran, B., Zhang, M., and Xue, B. A pso based hybrid feature selection algorithm for high-dimensional classification. In *2016 IEEE congress on evolutionary computation (CEC)*, pp. 3801–3808. IEEE, 2016.
- Van Rijn, J. N., Bischl, B., Torgo, L., Gao, B., Umaashankar, V., Fischer, S., Winter, P., Wiswedel, B., Berthold, M. R., and Vanschoren, J. Openml: A collaborative science platform. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part III 13*, pp. 645–649. Springer, 2013.
- Vanschoren, J., Van Rijn, J. N., Bischl, B., and Torgo, L. Openml: networked science in machine learning. *ACM SIGKDD Explorations Newsletter*, 15(2):49–60, 2014.



- Xiaojuan, L. and Yu, Z. A data integration tool for the integrated modeling and analysis for east. *Fusion Engineering and Design*, 195:113933, 2023.
- Yu, S., Wang, X., and Príncipe, J. C. Request-and-reverify: Hierarchical hypothesis testing for concept drift detection with expensive labels. *arXiv preprint arXiv:1806.10131*, 2018.
- Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., and Smola, A. J. Deep sets. *Advances in neural information processing systems*, 30, 2017.
- Zha, D., Bhat, Z. P., Lai, K.-H., Yang, F., Jiang, Z., Zhong, S., and Hu, X. Data-centric artificial intelligence: A survey. *arXiv preprint arXiv:2303.10158*, 2023.
- Zheng, L., Yuan, J., Wang, C., and Kong, L. Efficient attention via control variates. *arXiv preprint arXiv:2302.04542*, 2023.

## A. Benchmarking Methods

DataFix detects and localizes feature shifts using an iterative adversarial approach called DF-Locate. At each iteration, a random forest classifier is trained to distinguish between samples from a reference and a query distribution. The predicted class probabilities from this discriminator are used to compute the total variation distance (TVD), and its feature importance scores, based on the mean decrease of impurity, are used to locate the features originating the shift. These features are removed in successive rounds until the estimated divergence falls below a threshold or until half the features have been eliminated. A final refinement step uses a knee-detection algorithm to choose the optimal stopping point in the removal process. The selection of features is controlled by a dynamic threshold defined as the product of the TVD and a hyperparameter  $\tau$ .

Classical univariate statistical filters include Mutual Information statistics (MI) and selectKbest. These methods rank features based on their statistical association with the target. MI measures entropy shared between each feature and the target, while SelectKBest applies ANOVA-F for continuous features and Chi-squared for categorical ones. ANOVA-F tests variance for mean differences in continuous features across target classes, while the Chi-square test evaluates categorical feature-target dependency by comparing observed frequencies across target classes.

More advanced iterative selection methods include Minimum Redundancy Maximum Relevance (MRMR) and Fast Conditional Mutual Information Maximization (FAST-CMIM). These methods attempt to balance feature relevance with redundancy by considering mutual information conditioned on already selected features. MRMR iteratively selects relevant features while minimizing redundancy with the selected features so far. FAST-CMIM iteratively selects features that maximize MI, conditional to previously selected features. These methods, while more expressive than univariate tests, suffer from poor scalability and cannot process large datasets within practical time constraints.

Other benchmarked methods include model-based and statistical testing techniques specifically designed for feature shift localization. These include MB-SM (Multivariate Gaussian with Fisher-divergence test), MB-KS (Multivariate Gaussian with KS test), KNN-KS (K-Nearest Neighbors with KS statistic), and Deep-SM (deep density neural models with Fisher-divergence test). These approaches operate under stronger assumptions about data structure and require prior knowledge of the number of corrupted features, which is rarely available in real-world applications.

## B. Training and Validation Datasets

### B.1. OpenML Datasets

We source a diverse collection of datasets from OpenML (Van Rijn et al., 2013), selecting only those in a tabular format with at least 10 features and a minimum of 500 samples. To ensure data quality, we preprocess the datasets by removing constant features and addressing missing values using two strategies. In the first strategy, we remove features with more than 40% missing values, followed by the elimination of samples with missing data. In the second strategy, we remove features with more than 70% missing values before discarding samples with missing data. If both strategies preserve the required dimensions, we randomly select one for application. We also carefully remove any OpenML dataset that overlaps with the test datasets. For efficiency, large datasets exceeding 40M cell values are partitioned into ten equal-sized sets to accelerate processing. As a result, we obtain a total of 1,032 cleaned and partitioned datasets from OpenML.

### B.2. Algebraic Simulated Datasets

We generate 184 simulated datasets using Polynomial, Sine, and Logarithmic functions, each containing 5,000 samples and 1,000 features. Feature values from Polynomial functions are derived from  $f(x) = ax^4 + bx^3 + cx^2 + dx + e$ , with parameters  $a, b, c, d$ , and  $e$  uniformly sampled from  $[-50, 50]$ , and the degree randomly set to either 3 or 4. Sine functions follow  $f(x) = a \cdot \cos(bx + c)$ , where parameter  $a$  is sampled from  $[-50, 50]$ , and  $b$  and  $c$  are sampled from  $[-\pi, \pi]$ . Logarithmic functions are defined as  $\log_a(x + 6)$ , with base  $a$  sampled from  $[2, 10]$ . The input values,  $x$ , are drawn from the range  $[-5, 5]$ . Function parameters remain fixed across all samples.

### B.3. Probabilistic Simulated Datasets

We generate 184 simulated datasets based on probabilistic distributions, including Gaussian, Bernoulli, and Beta mixture models. Each dataset consists of 5,000 samples and 1,000 features. To construct these datasets, we begin by selecting a base distribution  $\mathcal{D}$  from {Gaussian, Bernoulli, Beta} and initializing parameters for a mixture model with a randomly determined

number of components  $K$ , where  $1 \leq K \leq 100$ . For the Gaussian distribution, the mean vector  $\mu_k = (\mu_{k1}, \mu_{k2}, \dots, \mu_{kd})$  is drawn from a standard normal distribution  $\mathcal{N}(0, 1)$ , while the variance vector  $\sigma_k^2 = (\sigma_{k1}^2, \sigma_{k2}^2, \dots, \sigma_{kd}^2)$  is sampled from a Uniform distribution  $\mathcal{U}(0.1, 1.1)$  for each feature independently. For the Bernoulli distribution, the probability vector  $p_k = (p_{k1}, p_{k2}, \dots, p_{kd})$ , which represents the probability of success for each feature, is sampled from a uniform distribution  $\mathcal{U}(0, 1)$ . For the Beta distribution, the parameters  $\alpha_k = (\alpha_{k1}, \alpha_{k2}, \dots, \alpha_{kd})$  and  $\beta_k = (\beta_{k1}, \beta_{k2}, \dots, \beta_{kd})$  are drawn from a uniform distribution  $\mathcal{U}(1, 2)$ . Samples are generated using the weighted sum of these mixture components, defined as  $\sum_{k=1}^K \pi_k \mathcal{D}(\theta_k)$ , where  $\pi_k$  represents the mixing coefficients, satisfying  $\sum_{k=1}^K \pi_k = 1$ .

Each dataset has a 25% chance of undergoing multiple transformations, with the number of transformation steps  $I$  randomly chosen between 1 and 5. These transformations include normalization (standard or min-max scaling), followed by a linear transformation in which the dataset is multiplied by a randomly sampled matrix  $W \in \mathbb{R}^{d \times d}$ . The entries of  $W$  are drawn from one of four distributions:  $\mathcal{U}(0, 1)$ ,  $\mathcal{N}(0, 1)$ , Beta(1, 1), or Bernoulli(0.5). Additionally, a non-linear transformation is applied, randomly chosen from the ReLU, GELU, sigmoid, hyperbolic tangent, or logarithmic transformation. To preserve structural information, residual connections are sometimes applied, where the transformed data is combined with its previous or initial state, either directly or after undergoing normalization.

## C. Testing Manipulation

Table 5 outlines the manipulation types used to induce shifts during evaluation. Some distort marginal distributions (E1, E2, E4, E5, E6, and E7), with manipulation E4 leaving the mean approximately unchanged. Others shuffle feature values across samples, altering feature correlations but not marginal distributions (E3 and E8). Manipulations E9 and E10 use KNN predictions to replace corrupted continuous and categorical features, respectively. For a more detailed explanation, see (Barrabés et al., 2023).

Table 5. Manipulation types applied to continuous and/or categorical features during benchmarking (Table from (Barrabés et al., 2023)).

Type	Mapping	Description	Shift	Data
E1	$x \sim \text{Uniform}(0, 1)$	Each value is substituted by a random number between 0 and 1.	$p_i \neq q_i$	Cont.
E2	$1 - x$	Each value is negated.	$p_i \neq q_i, \mathbb{E}[q_i] = 1 - \mathbb{E}[p_i]$	Both
E3	$P_i X_i$	$P_i$ is a random permutation matrix applied to feature $i$ .	$p_i = q_i, p_C \neq q_C,$ $q_C = \prod_{i \in C} q_i$	Both
E4.1-4.3	$\text{clamp}_{0,1}(x + \alpha\sigma)$ $\sigma \sim \text{Rademacher}(0.5)$	Add constant noise with a random sign. $\alpha \in \{0.02, 0.05, 0.1\}$ for 4.1-4.3 respectively.	$p_i \neq q_i, \mathbb{E}[p_i] \approx \mathbb{E}[q_i]$	Cont.
E5	$\text{round}(x)$	Values are binarized.	$p_i \neq q_i$	Cont.
E6.1-6.3	$b(1 - x) + (1 - b)x$ $b \sim \text{Bernoulli}(\rho)$	Values are negated with probability $\rho \in \{0.2, 0.4, 0.6\}$ for 6.1-6.3 respectively.	$p_i \neq q_i,$ $\mathbb{E}[q_i] = \rho + (1 - 2\rho)\mathbb{E}[p_i]$	Cat.
E7	$\text{MLP}(x)$	Forward through an MLP with min-max normalization or binarization.	$p_i \neq q_i$	Both
E8	$P X_i$	$P$ is a random permutation matrix applied to all features simultaneously.	$p_i = q_i, p_C = q_C, p \neq q$	Both
E9	$\text{KNN}(x)$	Predict feature with KNN (Regressor).	-	Cont.
E10	$\text{KNN}(x)$	Predict feature with KNN (Classifier).	-	Cat.



## D. Exploration of Attention Mechanisms

We investigated the integration of attention mechanisms across features within the residual blocks of both the Neural Embedding Network and the Prediction Network to effectively capture diverse data patterns and feature correlations. Upon incorporation, the attention mechanisms were coupled with MLPs implemented using convolutional layers with unit-sized kernels.

### D.1. Efficient EVA Attention Layers

Using full attention across features was infeasible for high-dimensional datasets due to its quadratic computational requirements. Consequently, we investigated the use of EVA attention layers (Zheng et al., 2023). Attention layers without positional embeddings are inherently equivariant; therefore, we did not include positional embeddings. However, efficient (approximate) attention layers, such as EVA, can fail to preserve equivariance under certain conditions.

### D.2. Sequence Handling and Window Size Adjustment

Key considerations involve ensuring that the input size to the EVA layer is not only smaller than the sequence length but also divisible by it. Managing variable-length sequences requires meticulous data processing, achieved through padding and the application of a key padding mask for efficient sequence handling. The window size  $w$  of the EVA attention was dynamically adjusted based on the sequence length to optimize processing efficiency and effectiveness, specifically determined as  $w = \min(\max(8, \lfloor \frac{d}{4} \rfloor), 32)$ .

### D.3. Limitations

Despite EVA attention layers being more efficient than traditional attention mechanisms (Liu et al., 2021), their integration into both the Neural Embedding Network and the Prediction Network resulted in CUDA out-of-memory issues for certain configurations. Consequently, we attempted to incorporate attention solely within the Prediction Network. While this approach alleviated memory issues, the performance remained below that of our baseline configuration without any attention mechanisms.

## E. Hyperparameter Tuning

### E.1. Training Setup and Hardware Specifications

We conducted a random search across various hyperparameters of the FSL-Net architecture and its training strategy. Training for each network was terminated if the validation loss did not improve for 50 consecutive evaluation intervals, with the maximum training duration limited to three days. To expedite the training process, a single NVIDIA-SMI GPU with 32GB of memory was used. The networks were validated on 50 validation datasets every 2,500 training iterations. The model checkpoint achieving the highest validation accuracy was selected as the optimal network and subsequently trained for up to seven days. This best-performing model was employed for inference and evaluation in the benchmarking experiments.

### E.2. Alternative Statistical Measures

In addition to the statistical measures delineated in the main text, we investigated the following metrics: skewness (indicating asymmetry), kurtosis (reflecting tail heaviness), index of dispersion (representing the variance-to-mean ratio), and trimmed mean deviation (a robust estimator of central tendency that excludes outliers). The formulas for each measure are presented in Table 6. These measures were ultimately excluded from the final analysis, as they did not confer any discernible advantage.

Table 6. Formulas for additional statistical measures of the Statistical Descriptor Network.  $N$  is the number of observations in  $X$ ,  $x_j$  represents the  $i$ th dimension of the  $j$ th sample (we skip the subscript  $i$  for brevity),  $x_{(l)}$  is the  $i$ th dimension of the  $l$ th sample after sorting the samples from smallest to largest,  $r$  is the count of observations trimmed from each end in trimmed mean deviation, and  $\bar{x}$  and  $\sigma$  are the empirical mean and standard deviation of the  $i$ th dimension.

Statistical Measure	Linear	Equation $\mu_{i,k}$
Skewness	Yes	$\frac{1}{N} \sum_{j=1}^N \left( \frac{x_j - \bar{x}}{\sigma} \right)^3$
Kurtosis	Yes	$\frac{1}{N} \sum_{j=1}^N \left( \frac{x_j - \bar{x}}{\sigma} \right)^4$
Index of Dispersion	Yes	$\frac{\sigma^2}{\bar{x} + \epsilon}$
Trimmed Mean Deviation	No	$\frac{1}{N-r} \sum_{j=r+1}^{N-r} x_{(j)}$

### E.3. Alternative Merging Operations

In addition to the normalized squared difference discussed in the main text, we examined several alternative merging operations within the Prediction Network. The following methods were evaluated; however, none yielded performance improvements over the existing approach: concatenation  $\alpha(\mu_p, \mu_q) = [\mu_p; \mu_q]$ ; element-wise difference  $\alpha(\mu_p, \mu_q) = \mu_p - \mu_q$ ; and squared difference  $\alpha(\mu_p, \mu_q) = (\mu_p - \mu_q)^2$ . Note that, for the concatenation method, the merged feature vector has dimensions  $\mu_{p,q} \in \mathbb{R}^{d \times 2t}$ , whereas for the other merging operations, it has dimensions  $\mu_{p,q} \in \mathbb{R}^{d \times t}$ .

### E.4. Hyperparameter Search

#### E.4.1. NETWORK TUNING PARAMETERS

Table 7 provides an overview of the search spaces and the optimal values determined for each network-related hyperparameter in FSL-Net. These parameters encompass configurations for the statistical measures, Moment Extraction Network, Neural Embedding Network, and Prediction Network. For parameters not explicitly specified, default values were applied.

#### E.4.2. ATTENTION TUNING PARAMETERS

Table 8 presents the hyperparameters of the attention mechanism explored during the optimization process. These parameters regulate the EVA attention layers within FSL-Net, including the number of heads, the number of landmarks, the window factor, and dropout rates.

#### E.4.3. OPTIMIZATION TUNING PARAMETERS

Table 9 outlines the search space and optimal values for optimization hyperparameters in FSL-Net’s training strategy. This includes the loss function and Adam optimizer settings.

Table 7. Search space and optimal values for tuned network hyperparameters in FSL-Net.

Component	Hyperparameter	Possible Values	Optimal Value
<b>Statistical Measures</b>	Mean	{True, False}	<b>True</b>
	Standard Deviation	{True, False}	<b>True</b>
	Median	{True, False}	<b>True</b>
	Mean Absolute Deviation	{True, False}	<b>True</b>
	$p$ -order Moments	{True, False}	<b>True</b>
	$p$	{2}, {3}, {2, 3}	<b>2, 3</b>
	Histogram	{True, False}	<b>True</b>
	Empirical CDF	{True, False}	<b>True</b>
	# Bins	{100}, {50, 100}	<b>100</b>
	Skewness	{True, False}	<b>False</b>
	Kurtosis	{True, False}	<b>False</b>
	Index of Dispersion	{True, False}	<b>False</b>
	Trimmed Mean Deviation	{True, False}	<b>False</b>
	Trimmed Percentage	{0.1}	<b>0.1</b>
<b>Moment Extraction Network</b>	# Kernels	{32, 64, 128}	<b>64</b>
	Kernel Size	{75, 125}	<b>75</b>
	Dilation	{1}	<b>1</b>
	Activation	{ReLU, Tanh}	<b>ReLU</b>
<b>Neural Embedding Network</b>	# Residual Layers	{3, 5, 7}	<b>5</b>
	# Kernels	{32, 64}	<b>64</b>
	Kernel Size	{5, 7}	<b>5</b>
	Dilation	{1}	<b>1</b>
	Activation	{Tanh, GELU}	<b>Tanh</b>
<b>Prediction Network</b>	# Residual Layers	{3, 5, 7}	<b>7</b>
	# Kernels	{32, 64}	<b>64</b>
	Kernel Size	{5, 7}	<b>5</b>
	Dilation	{1}	<b>1</b>
	Activation	{Tanh, GELU}	<b>Tanh</b>
	Combination	{Concatenation, Element-wise difference, Squared difference, Normalized squared difference}	<b>Normalized squared difference</b>

Table 8. Search space and optimal values for tuned attention hyperparameters.

Component	Hyperparameter	Possible Values	Optimal Value
<b>Attention</b>	# Heads	{4, 8}	<b>4</b>
	# Landmarks	{8}	<b>8</b>
	Window factor	{4}	<b>4</b>
	Window size	{7, 9}	<b>7</b>
	Overlapping windows	{True, False}	<b>False</b>
	Attention dropout	{0, 0.3, 0.5}	<b>0</b>
	Projection dropout	{0}	<b>0</b>

Table 9. Search space and optimal values for optimization hyperparameters in the training strategy.

Component	Hyperparameter	Possible Values	Optimal Value
<b>Loss Function</b>	$\lambda$	{0.0001, 0.001}	<b>0.001</b>
<b>Adam Optimizer</b>	Learning Rate	{0.0001, 0.001, 0.01}	<b>0.001</b>
	Learning Rate Gamma	{0.9, 0.999, 0.9995}	<b>0.9995</b>

## F. Extended Experimental Results

### F.1. Median-based Feature Shift Localization Performance and Runtime Comparison

Figure 3 presents a median-based analysis of feature shift localization performance and runtime, complementing the mean-based evaluation in the main text. For each method, F-1 scores and runtimes are first averaged across manipulation types and fractions of manipulated features, then aggregated using the median across datasets. This approach eliminates the need for imputation in cases where slower methods fail to complete for some datasets, enabling fairer comparisons across a broader set of methods. However, the mean-based results remain more representative for consistently successful approaches like FSL-Net and DataFix, which completed all evaluations. Figure 3a summarizes the resulting median F-1 scores, while Figure 3b and Figure 3c visualize the trade-off between performance and computational cost, plotting median F-1 scores against median and maximum runtime, respectively. In these plots, higher positions indicate better localization accuracy, while positions further to the left denote lower runtime. These comparisons again confirm that FSL-Net matches DataFix in localization accuracy while offering a substantial advantage in runtime efficiency. Despite being only slightly slower than SelectKBest, FSL-Net far exceeds this method in accuracy. Meanwhile, methods such as MRMR and FAST-CMIM continue to struggle with scalability and performance under median-based evaluation. MB-SM, MB-KS, KNN-KS, and Deep-SM show higher F-1 scores when evaluated by the median rather than the mean, with KNN-KS performing the best among them. However, all of these methods still lag behind FSL-Net, despite benefiting from access to the ground truth  $|C|$  – a condition rarely met in practice – and suffer from extremely poor scalability on large datasets. Figure 3d plots the mean runtime against the product of sample and feature sizes for each dataset. FSL-Net maintains a significantly lower mean runtime than DataFix across datasets of varying dimensionality. Figure 3e presents median F-1 scores by manipulation type, while Figure 3f shows mean F-1 scores by dataset. FSL-Net matches or surpasses the performance of competing methods across nearly all manipulation types, with two exceptions: E9, where DataFix achieves higher performance, consistent with mean-based results, and E4, where both DataFix and KNN-KS exhibit superior accuracy. Notably, FSL-Net demonstrates a clear advantage on high-dimensional datasets such as Phenotypes, Founders, and Canine, outperforming all baselines and highlighting its effectiveness in scenarios involving large feature sets.



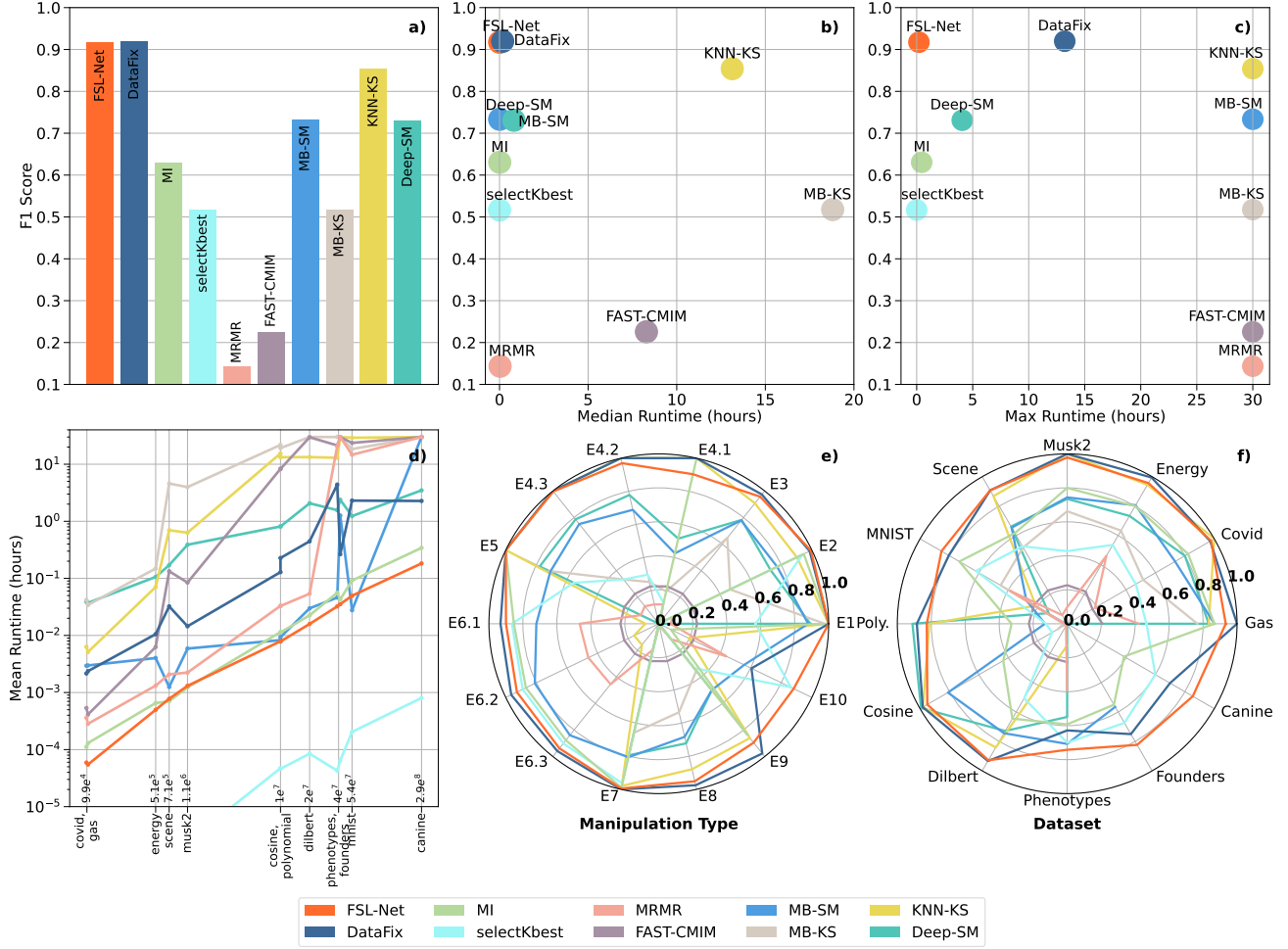


Figure 3. Performance and runtime comparison across feature shift localization methods: a) median F-1 scores across manipulation types, fractions of manipulated features, and datasets; b) median F-1 scores vs. median runtime; c) median F-1 scores vs. maximum runtime; d) mean runtime vs. sample-feature size product per dataset; e) median F-1 scores by manipulation type; f) mean F-1 scores by dataset.

## F.2. Performance and Computational Efficiency of FSL-Net and DataFix

Figure 4 displays the mean F-1 scores (top), mean runtimes (middle), and maximum runtimes (bottom) for both FSL-Net and DataFix, computed across various manipulation types and fractions of manipulated features. FSL-Net exhibits comparable feature shift localization performance to DataFix on the majority of datasets, with a notable advantage on large datasets such as Phenotypes, Founders, and Canine, while also demonstrating significantly greater computational efficiency. The pronounced disparity between the mean and maximum runtimes of DataFix on certain datasets suggests that its processing time is highly sensitive to the complexity of the shifts, potentially necessitating multiple iterations to achieve convergence. In contrast, FSL-Net executes only a single forward pass through the network, ensuring scalability to both high-dimensional and large datasets.

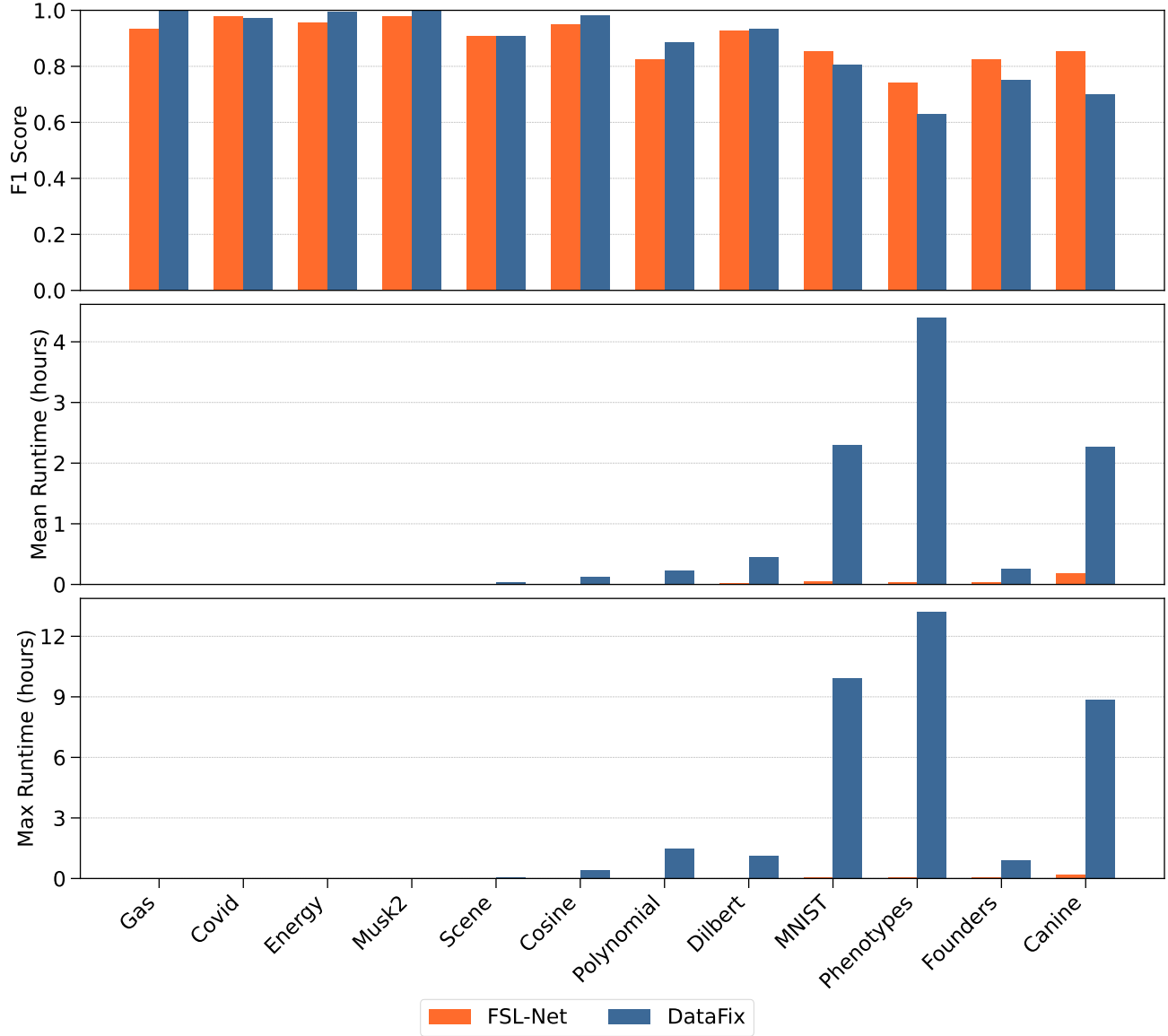


Figure 4. Mean F-1 scores (top), mean runtimes (middle), and maximum runtimes (bottom) of FSL-Net and DataFix by dataset.

### F.3. Extended Evaluation of FSL-Net and DataFix on CIFAR10 and COIL-100

Table 10 presents an extended evaluation of FSL-Net and DataFix on two image datasets: CIFAR10 (10k samples) and COIL-100. We report the mean F-1 score in feature shift localization and the mean and maximum runtime (in hours), computed across manipulation types and fractions of manipulated features. FSL-Net achieves higher F-1 scores on average and consistently outperforms DataFix in terms of computational efficiency, exhibiting significantly lower runtimes in both average and worst-case scenarios.

Table 10. Comparison of FSL-Net and DataFix averaged across manipulation types and fractions of manipulated features on two datasets: CIFAR10 (10k) and COIL-100.

Dataset	F-1 Score		Mean Runtime (hours)		Max Runtime (hours)	
	FSL-Net	DataFix	FSL-Net	DataFix	FSL-Net	DataFix
<b>CIFAR10</b>	<b>0.9565</b>	0.8911	<b>0.0319</b>	0.2948	<b>0.0336</b>	0.8084
<b>COIL-100</b>	0.9720	<b>0.9805</b>	<b>0.4906</b>	2.9228	<b>0.8406</b>	9.1567

#### E.4. Runtime Improvement of FSL-Net over DataFix

Figure 5 presents the mean runtime improvement of FSL-Net over DataFix across datasets, sorted by increasing dataset size (measured as the product of the number of samples and the number of features). The speedup factor – defined as the ratio of DataFix’s runtime to FSL-Net’s – measures the performance gain, with higher values indicating greater efficiency. The results are averaged across manipulation types and fractions of manipulated features. On average, FSL-Net achieves a substantial speedup of  $35.8\times$  (indicated by the dashed line), consistently outperforming DataFix across datasets. Among all datasets, the high-dimensional Phenotypes dataset exhibits the greatest speedup, with FSL-Net outperforming DataFix by a remarkable  $136.3\times$ .

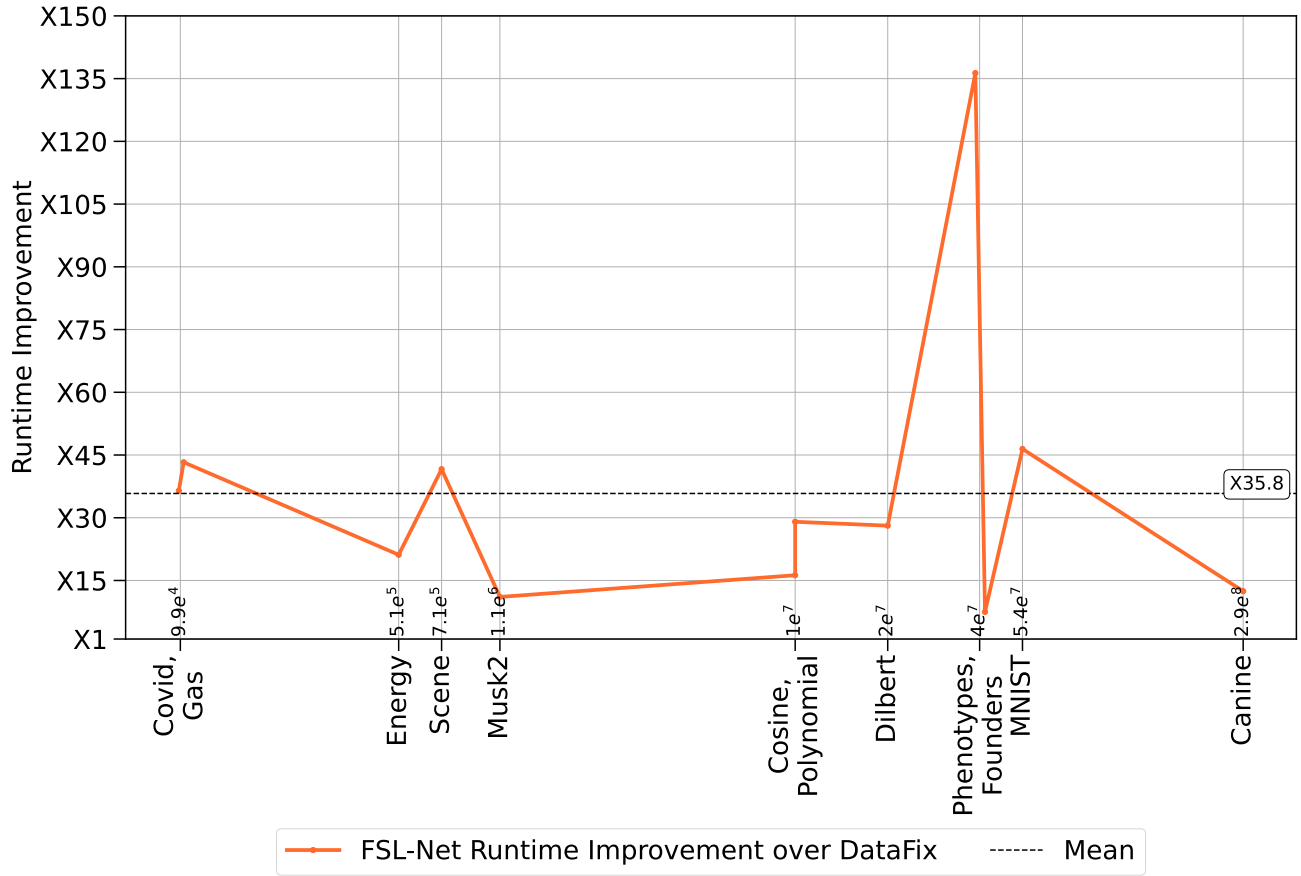


Figure 5. Runtime improvement of FSL-Net over DataFix across datasets, sorted by dataset size (number of samples  $\times$  number of features). The speedup factor indicates how many times faster FSL-Net is compared to DataFix. The dashed line marks the average speedup of  $35.8\times$ .



### F.5. Performance of the Threshold-Based Variant of the SM-only Baseline

We evaluate a simplified (though constrained) configuration of FSL-Net that eliminates the Prediction Network and instead relies solely on thresholding the combined statistical measures, computed as the normalized squared differences between the reference and query. In this threshold-based approach, the statistical measures are reduced to a scalar score by taking either the mean or the maximum of these differences. Figure 6 shows the mean F-1 scores for the SM-only variant of FSL-Net, where feature shift localization is performed entirely through thresholding. The F-1 scores are averaged across datasets, manipulation types, and fractions of manipulated features. The best performance is achieved using the mean-based method with a threshold of 0.002, yielding an optimal F-1 score of 0.307.

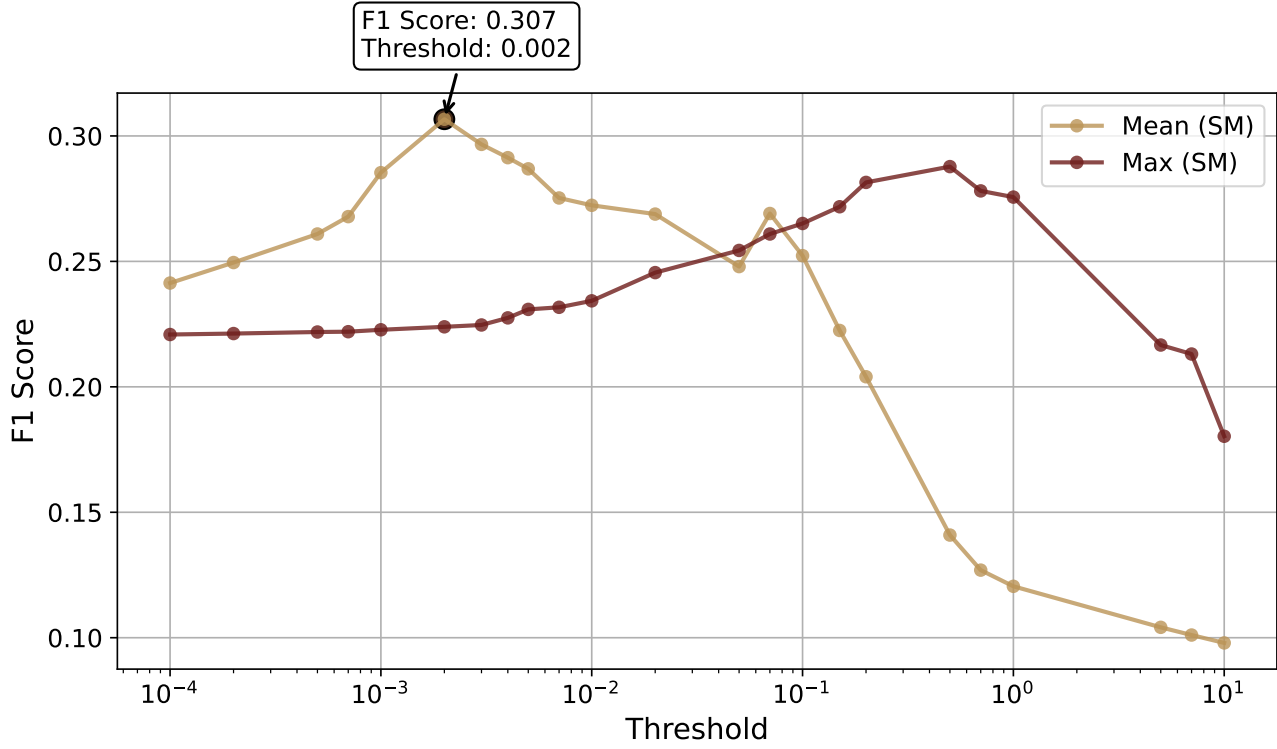
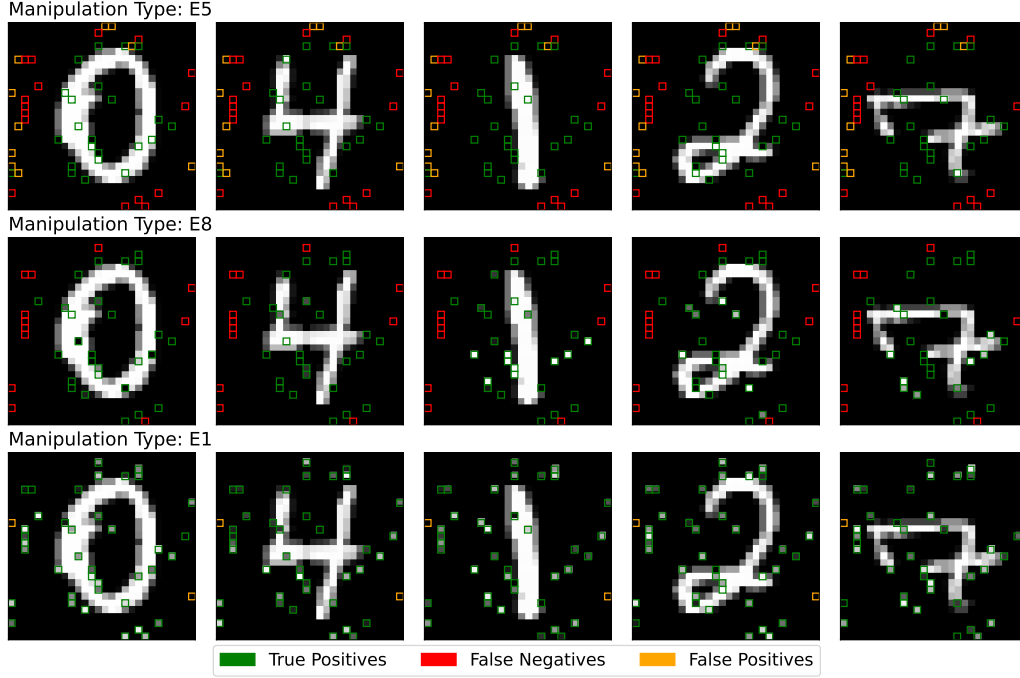


Figure 6. Mean F-1 scores of the SM-only variant of FSL-Net, where the Prediction Network is omitted and feature shift localization relies solely on thresholding the statistical measures. The F-1 scores are averaged across datasets, manipulation types, and fractions of manipulated features. Statistical measures are reduced to scalar values taking either the mean or maximum of the normalized squared differences between the reference and the query.

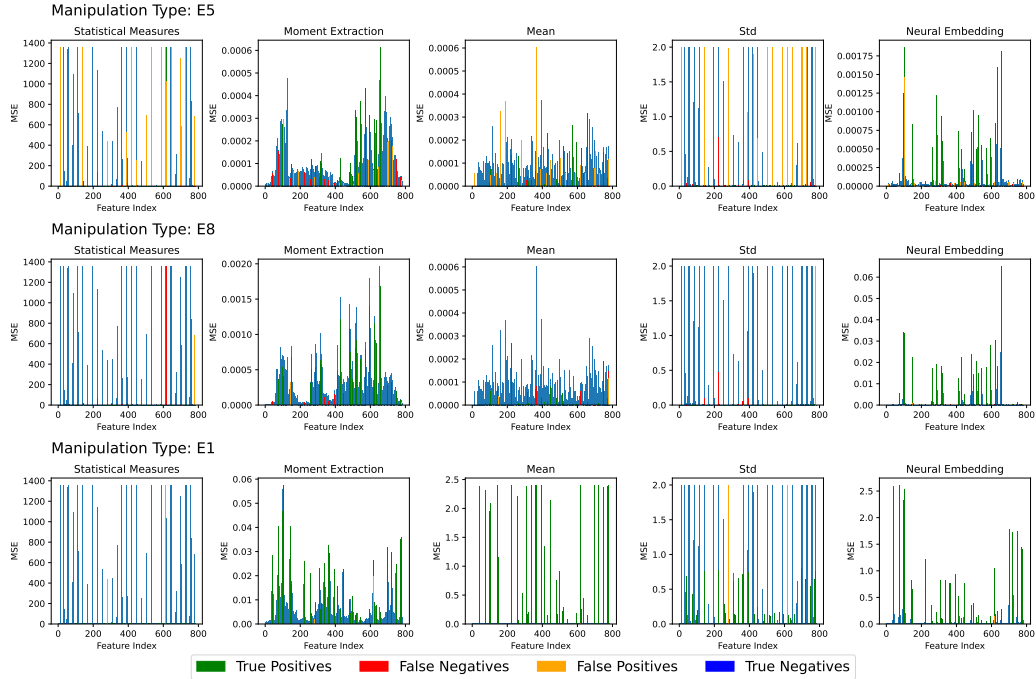
### E.6. Qualitative Evaluation of FSL-Net on the MNIST Dataset

Figure 7 provides a qualitative assessment of the performance of FSL-Net on the MNIST dataset under various manipulation types. The first subfigure (Figure 7a) illustrates the ability of FSL-Net to identify manipulated features in five sample images. Each row represents a distinct manipulation type applied to modify 5% of the image features (i.e., pixels), with the top row corresponding to E5, the middle row to E8, and the bottom row to E1. Correctly detected manipulations are highlighted in green, while undetected modifications (false negatives) and misclassified pixels (false positives) are colored red and orange, respectively. The results indicate that FSL-Net effectively detects manipulation E1, while its performance is less consistent for manipulations E5 and E8. The difficulty in detecting E5 comes from its binary rounding of pixel values to 0 or 1, often blending seamlessly into the digit structure. Similarly, E8, which randomly permutes pixel values, disrupts spatial coherence without necessarily creating visually distinct artifacts, making detection more challenging.

The second subfigure (Figure 7b) presents the Mean Squared Error (MSE) between the reference and query statistical functional maps derived from statistical measures (including the mean and standard deviation), the Moment Extraction Network, and the Neural Embedding Network. Each bar represents a distinct feature, with color coding indicating classification outcomes: true positive (green), false negative (red), false positive (orange), and true negative (blue). Manipulation E1 (bottom row) is easily detectable, exhibiting high MSE values, particularly in the statistical functional maps derived from the mean, the Moment Extraction Network, and the Neural Embedding Network. In contrast, more subtle manipulations, such as E5 (top row) and E8 (middle row), display smaller MSE values between reference and query for certain features, making their detection more challenging. Still, several features are correctly classified for these challenging manipulations, with the Neural Embedding Network and the Moment Extraction Network producing the most distinguishable statistical functional maps. The mean and standard deviation measures are ineffective in detecting manipulation E8, as it only affects correlation, with low MSE values for these measures.



(a) Comparison of FSL-Net output for four examples from the MNIST dataset, with manipulation type E5 (top row), E8 (middle row), and E1 (bottom row). In each case, 5% of the image features were manipulated. Green rectangles indicate correctly corrupted features, red rectangles denote false negatives, and orange rectangles represent false positives.



(b) Mean Squared Error (MSE) between reference and query statistical functional maps derived from statistical measures, the Moment Extraction Network, and the Neural Embedding Network. Each bar represents a feature, with colors indicating classification outcomes. The bottom row corresponds to manipulation type E1, while the middle and top rows correspond to the more challenging manipulation types E8 and E5, respectively.

Figure 7. Qualitative evaluation of FSL-Net on the MNIST dataset.