SENSITIVITY BASED MODEL AGNOSTIC SCALABLE EXPLANATIONS OF DEEP LEARNING

A PREPRINT

Manu Aggarwal*
National Institutes of Health
Bethesda, MD
manu.aggarwal@nih.gov

Nick Cogan
Department of Mathematics
Florida State University
Tallahassee, FL
cogan@math.fsu.edu

Vipul Periwal
National Institutes of Health
Bethesda, MD
vipulp@niddk.nih.gov

February 21, 2025

ABSTRACT

Machine learning using deep neural networks (DNNs) has become ubiquitous in data-driven predictive learning. However, their complex architecture often obscures what they have learned from the data. This information is crucial to validate these models when their predictions can affect human life, for example, biological and clinical predictions. We design SensX, a model agnostic explainable AI (XAI) framework to explain what a trained DNN has learned. We introduce the notion of justifiable perturbations to systematically conduct global sensitivity analysis. Benchmarks using synthetic data sets show that SensX outperformed current state-of-the-art XAI in accuracy (up to 50% higher) and computation time (up to 158 times faster), with higher consistency in all cases. Moreover, only SensX scaled to explain vision transformer (ViT) models defined for input images with more than $150,000$ features. SensX validated the ViT models by showing that the features they learn as important for different facial attributes are intuitively accurate. Further, SensX revealed that there may be biases inherent to the model architecture, an observation that is possible only when the model is explained at the full resolution of the input image. Finally, we use SensX to explain a DNN trained to annotate biological cell types using single-cell RNA-seq data sets with more than a million cells and more than $56,000$ genes measured per cell. SensX determines the different sets of genes that the DNNs have learned to be important to different cell types.

Keywords explainable AI, global sensitivity analysis

1 Introduction

Deep neural network models (DNNs) are powerful tools for learning patterns in large data sets to make predictions with high accuracy. Further, formulation of DNNs does not require any explicit system-specific rules or constraints to be imposed a priori and can be purely data driven. This makes DNNs indispensable when there is a lack of human knowledge of the system-specific mechanistic and functional relations. The success of DNNs is theoretically assured by their architecture of interconnected networks with a sufficiently large number of parameters [Hornik et al., 1989], which often ranges from millions to trillions in practice. The complexity of the interconnectedness of such a large number of parameters often obscures explanation of the mechanistic relations that a trained DNN has learned as the criteria for its predictions. However, this is crucial when they are used to make biological, and especially clinical, predictions. For example, DeGrave et al. [2021] showed that DNNs trained to detect COVID-19 in chest radiographs relied on spurious factors rather than medical pathology. Consequently, these models failed in new hospitals even though they had high accuracy on test data. Besides conceptual validation of what a DNN has learned, another important reason for explaining DNNs is that insights into functional relations is a critical step in the scientific method of developing hypotheses and designing experiments to further our scientific knowledge.

*Corresponding author

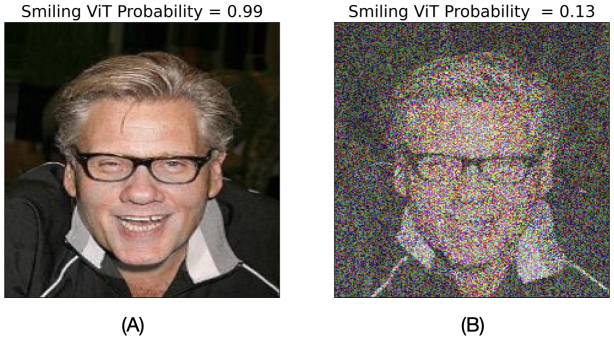


Figure 1: XAI based on global sensitivity analysis: (A) Vision transformer (ViT) trained to identify smiling human faces predicts that the face in the image is 'Smiling' with 99% confidence. (B) Perturbing features (all pixels across three color channels) reduces the model's predicted confidence of 'Smiling' to 13%. *To perturbations in which input features is this change in the model confidence sensitive to?*

The field of research to understand the mechanisms underlying DNNs has been referred to as explainable AI (XAI) [Gunning and Aha, 2019, Arrieta et al., 2020, Linardatos et al., 2020] and also as model interpretability [Lipton, 2018]. However, there is no consensus on the definitions of explainability and interpretability [Lipton, 2018, Arrieta et al., 2020]. We adopt the broad definition of explainability proposed in Arrieta et al. [2020]—explainability is to produce post-hoc details or reasons of a trained model to make its functions clear and easy to understand. In this regard, we first illustrate what our framework explains about a model.

A DNN trained to identify smiling human faces predicts with a 99% confidence that the face shown in Figure 1A is smiling. The DNN is the deterministic mathematical model, the model input is the image of size 224 pixels by 224 pixels by 3 color channels, and the corresponding model output is the predicted confidence that the face in the image is smiling. We say that the number of features of the input or the number of dimensions of the input space is $224 \times 224 \times 3 = 150,528$. A random perturbation of all features results in a new image (Figure 1B) for which the model predicts smiling with a reduced confidence of 13%. *Perturbations in which features of the input were important to change the model's prediction?* Our framework ranks the importance of features of a given input based on the sensitivity of the model output to perturbations in the features of the input.

This approach to XAI comes under the umbrella of sensitivity analysis (SA). In general, SA determines the effects of perturbations in input features on a scalar quantity of interest (QOI) that is a mathematical function of the features. SA methods are classified as local and global SA. Local SA considers perturbations in one input feature at a time keeping the rest fixed. Global SA (GSA) is a more comprehensive analysis because it considers simultaneous perturbations in all input features. However, GSA incurs a high computational cost. For example, Sobol' indices [Sobol, 2001] are popular GSA measures [Tosin et al., 2020], but their computational cost limits their application to a few tens of features [Ikonen, 2016]. Van Stein et al. [2022] compare computational feasibility of different GSA methods for XAI (GSA-XAI), but the number of features was limited to less than 100. In comparison, the example shown in Figure 1 has hundreds of thousands of features. Nevertheless, Paleari et al. [2021] and Van Stein et al. [2022] empirically showed that the Morris method is the least computationally costly GSA method among those tested, without compromising reliability and accuracy.

Besides the concern of computational scalability, a conceptual concern that we claim to be fundamental to GSA-XAI is to justify the domain of perturbations. To explain why this is crucial, we introduce three criteria in this work to define justifiable perturbations as follows. First, we say that the perturbations should be contextually meaningful. For example, if the input features are gene expression values of different genes, then GSA based on perturbations that result in biologically infeasible gene expression values (negative, arbitrarily large, etc.) might not be reasonable. This is particularly important for XAI since many black-box DNNs are inherently mathematically defined for all real numbers and will generate an output given any input. Second, the perturbations should have a significant effect on the model output. Otherwise, the resulting GSA measures are not relevant and can be misleading. We denote these by significant perturbations. Third, since we attribute importance to features for a given model input, the perturbations should be with respect to that input. We denote these by grounded perturbations. We next introduce our framework and clarify grounded perturbations.

We design SensX, a model agnostic XAI framework that conducts GSA using justifiable perturbations to explain model inference for a given input. Our GSA is based on the Morris method which we briefly outline follows. A set of simultaneous perturbations in all features can be visualized as a trajectory in the input space, perturbing exactly one

of the features along each step of the trajectory. Gradients are computed at each step using finite differences. When repeated for multiple sets of simultaneous perturbations, we get distributions of gradients with respect to features. Morris defined the mean and standard deviation of these distributions as linear and non-linear/interaction elementary effects of features, respectively. We implement and modify the Morris method as follows. First, the framework computes an input-specific domain of justifiable perturbations. We show later in one of our case studies that the input-specific domain might be much smaller than the global domain of the input space. This can potentially increase the accuracy of the results for the same number of model simulations since the elementary effects are estimated by sampling the domain. Second, the sets of simultaneous perturbations are generated such that the corresponding trajectories always start at the given model input. We say that they are *grounded* at a fixed reference point. In this work, we compute the mean of the distributions of absolute values of the gradients as suggested by [Campolongo et al., 2007] to get a single measure of importance for each feature. However, linear and nonlinear effects as proposed in the traditional Morris method can be computed without significant computational overhead because no additional model simulations are required.

Out of the vast repository of XAI methods [Linardatos et al., 2020], we compared SensX with SHapley Additive exPlanations (SHAP) [Lundberg, 2017] because it is the most widely used model agnostic XAI method ($31,309$ citations according to Google Scholar, February 2025). Furthermore, it theoretically unifies multiple other methods, including LIME [Ribeiro et al., 2016] and DeepLIFT [Shrikumar et al., 2017]. Two other model agnostic XAI methods that we considered but did not pursue are L2X [Chen et al., 2018] and TERP [Mehdi and Tiwary, 2024]. We considered L2X because it was shown to outperform SHAP in accuracy and computation time for synthetic data sets. However, a required hyperparameter k of the method is defined as the number of important features, and its known true value was chosen to benchmark synthetic data sets in Chen et al. [2018]. It was not clear to us how the choice of k will affect the accuracy and computation time when its true value is not known, especially when there are hundreds of thousands of features. Arguably, L2X could be carried out for a set of chosen values of k when its true value is unknown, but we do not know how to decide on an optimal value of k based on the results of the different choices. We considered TERP because of its recency and unique explanations based on thermodynamics. However, preliminary simulations showed a higher computation cost and lower accuracy than SHAP for synthetic data sets.

We show three case studies. (1) We benchmark SensX and SHAP using four synthetic data sets. (2) We explain vision transformer (ViT) models that are trained to identify specific facial attributes in images of human faces. (3) We train feedforward neural networks (FNNs) for automated cell annotation using single-cell RNA-seq data sets and determine the sets of genes they have determined to be important to identify different cell types.

2 Results

We show results from three different case studies using different model architectures. (1) FNNs for synthetic data sets labeled 0 or 1, (2) vision transformer (ViT) models for images of human faces annotated for smiling and eyeglasses, and (3) FNNs for single-cell RNA-seq data annotated with 21 different cell types. All models are binary classifiers that are trained on labeled data. For multi-class data sets in (2) and (3), binary classifiers are trained separately to identify each class. The QOI for GSA of each model is the confidence predicted by the model that the input belongs to the class (predicted probability of labeling the input 1). We will refer to this as the model confidence.

2.1 Benchmarking SensX and SHAP

We benchmark SensX and SHAP for four synthetic data sets—XOR, orange skin, nonlinear additive model, and switch [Chen et al., 2018, 2017]. Binary-labeled ten-dimensional samples (or model inputs) were generated for each data set according to the criteria detailed in Methods. The exact set of relevant features is known for each data set from the labeling criteria. The number of relevant features is $2, 4, 4,$ and 5 for XOR, orange skin, nonlinear additive model, and switch, respectively. To highlight the input-specific analysis of both methods, the set of five relevant features can be different for different samples in the switch data set.

We computed the SensX and SHAP values for 1000 test samples and ranked the features accordingly for each sample. To compare accuracy, we define top-k accuracy as the percentage of the test samples in which the top k-ranked features belong to the set of relevant features. We implemented both methods for different sets of hyperparameters. We also compared consistency of the accuracy of these heuristic methods by repeating computations with 100 different random seeds for each set of hyperparameters.

2.1.1 SensX achieves up to to 52% higher median accuracy

We computed the median accuracy across the 100 different runs for each set of hyperparameters (see supplementary). Table 2 compares the results for the best median accuracy achieved by the methods. SensX has higher median accuracy

model	top-k	median accuracy (%)		accuracy differential (%)	mean time taken (sec)		speed-up factor	CQD (lower is better)	
		SHAP	SensX		SHAP	SensX		SHAP	SensX
XOR	1	99.2	100	0.8	296	14	21	0.0015	0
XOR	2	97.9	100	2.1	2190	14	156	0.002	0
orange_skin	1	100	100		2206	14	158	0	0
orange_skin	2	99.8	100	0.2	2206	14	158	0.0005	0
orange_skin	3	98.7	100	1.3	2206	14	158	0.001	0
orange_skin	4	89	99.8	10.8	2206	14	158	0.0039	0
nonlinear_additive	1	100	100		2210	14	158	0	0
nonlinear_additive	2	99.7	100	0.3	2210	14	158	0.001	0
nonlinear_additive	3	96	100	4	2210	14	158	0.0052	0
nonlinear_additive	4	72	94.3	22.3	2210	17	130	0.0254	0.0085
switch	1	88.5	99.9	11.5	296	14	21	0.0479	0
switch	2	50.5	99.5	49	296	14	21	0.1089	0.0005
switch	3	16.3	67.6	51.3	296	14	21	0.2183	0.0037
switch	4	7.2	59.2	52	2218	14	158	0.049	0.0025
switch	5	2.4	12	9.6	1123	9	125	0.125	0.0661

Figure 2: Benchmarks on synthetic data sets to compare SHAP and SensX for accuracy, computation time, and consistency. Consistency is compared using coefficient of quartile deviation (CQD), rounded to four decimal places.

than SHAP in all cases, and also achieves a 100% accuracy in 8 cases whereas SHAP does so in 2 cases. Significant improvements are observed in nonlinear additive and switch data sets, in which SensX achieves up to 20% and 52% higher median accuracies than SHAP, respectively.

2.1.2 SensX is up to 158 times faster in run-time

We compare the mean computation times for the 100 different runs of both methods with the sets of hyperparameters that gave the best median accuracy. Table 2 shows that SensX is up to 158 times and at least 21 times faster than SHAP for this analysis.

2.1.3 SensX achieves higher consistency in all cases

We compare the consistency of the two methods by comparing the coefficients of quartile deviation (CQD) [Botta-Dukát, 2023] of accuracy from 100 runs with the sets of hyperparameters that gave the best median accuracy. A lower CQD implies higher consistency. Table 2 shows that SensX is more consistent than SHAP in all cases.

2.2 SensX validates vision transformer (ViT) models by explaining the characteristics of human facial attributes they have learned at full image resolution

Vision transformer (ViT) models [Dosovitskiy, 2020] apply the transformer architecture characterized by self-attention layers [Vaswani, 2017] to learn classification of images. We trained a ViT model on the publicly available Large-scale CelebFaces Attributes (CelebA) data set [Liu et al., 2015] to predict 'Not Smiling' and 'Smiling' for human faces. We denote this model by smiling-ViT. Similarly, we computed a ViT model that predicts eyeglasses on human faces, henceforth referred to as eyeglasses-ViT.

2.2.1 SensX scales to more than hundreds of thousands of features.

The input image has dimensions 224×224 pixels for each of the three color channels (RGB), totaling $150,528$ features. We were unable to use SHAP because it required at least 33 TB of memory to analyze a single test image, which was beyond the scope of the computational resources available to us. SensX takes around 10 GB of memory and 10 hours on a NVIDIA A100 GPU to process a single image.

2.2.2 SensX reveals relevant features at full resolution.

Figure 3 shows the SensX values of features as heat maps, separately for the three color channels of a test image. It is visually evident that the pixels with high SensX values are around the mouth for the smiling-ViT. In contrast, pixels near the bridge of the nose and the frame of the eyeglasses have high SensX values with respect to the eyeglasses-ViT. Hence, our framework reveals that smiling-ViT and eyeglasses-ViT have learned intuitively correct facial attributes for smiling and eyeglasses, respectively, of this image.

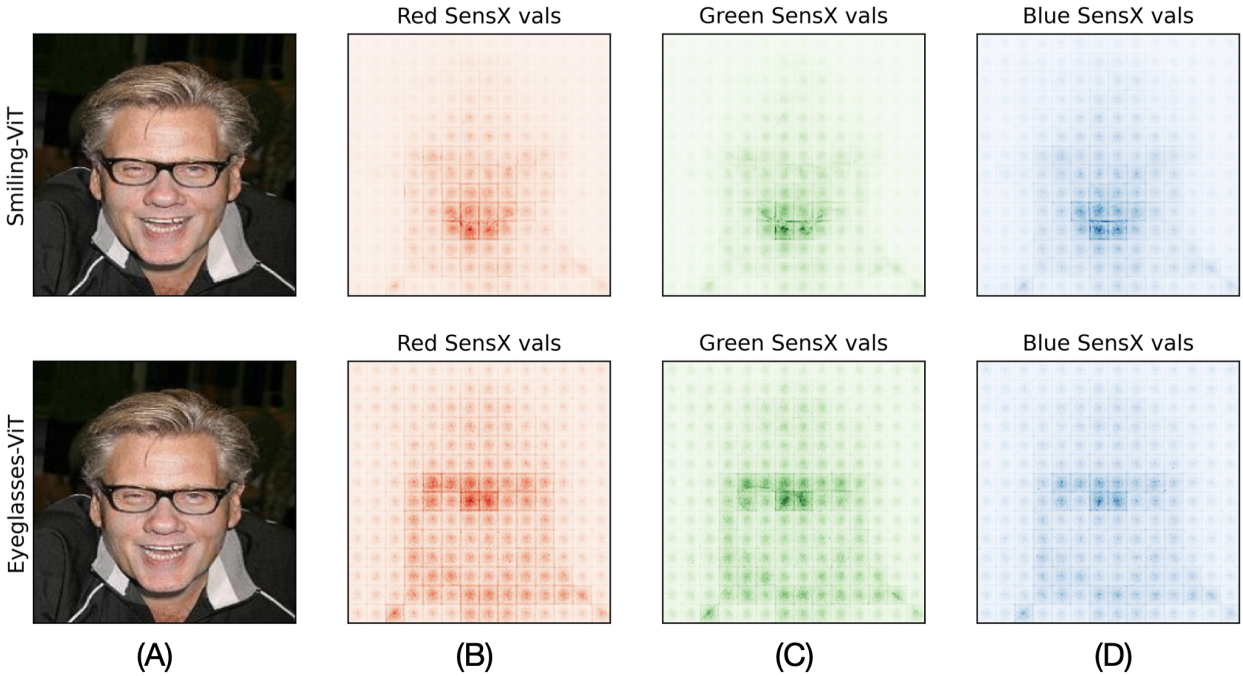


Figure 3: (A) Test image. (B)-(D) Heatmaps of SensX values of pixels for red, green, and blue channels for smiling-ViT (top) and eyeglasses-ViT (bottom). Pixels around the mouth have high SensX values for the smiling-ViT, whereas pixels at the bridge of the nose and frame of the eyeglasses have high SensX values for the eyeglasses-ViT.

2.2.3 Top ranked SensX features show an intuitively reasonable mask for smiling and eyeglasses learned by the ViT models.

To see the mask of the test image defined by top k SensX features, we define a new image with top k SensX features set to their values in the test image and the remaining features set to 0. Figure 4 shows the resulting images for $k = 1000, 2500, 5000$, and 10000 for both smiling- and eyeglasses-ViT. It is visually evident that the top k SensX features of smiling-ViT define a mask of the test image around the mouth and capture the smile, while the top k SensX features of eyeglasses-ViT capture the bridge of the nose and the frame of the eyeglasses. Hence, the top SensX features create human interpretable masks of the test image that reasonably conform with what smiling- and eyeglasses-ViT should consider as important. *Can we determine an optimal value of k?* To answer this, we first introduce the SensX landscape.

2.2.4 SensX landscape defines multiple characteristics of the model.

We conduct a more comprehensive analysis by perturbing the top k SensX features for a wider range of values of k and also for different values of the perturbation factor. The perturbation factor, denoted by $\delta \in [0, 1]$, defines the local neighborhood of the input sample to generate perturbations (see Methods). When $\delta = 0$, the local neighborhood is the singleton set of the input sample. When $\delta = 1$ it is the global domain. For each pair of values of k and δ, we compute the median of the model confidence for 1000 perturbations of the test image. The results are shown as interpolated heatmaps that we call SensX landscapes. Figure 5A shows SensX landscapes of the smiling- and eyeglasses-ViT for the test image. We make the following observations. (1) δ has to be greater than 0.2 and 0.5 to affect the median model confidence of the smiling- and eyeglasses-ViT models, respectively. (2) There exist distinct regions in the landscapes that result in a minimal value of the median model confidence. Importantly, the model confidence achieves its lowest value at k less than the total features. In other words, SensX landscape reveals a subset of features perturbing which reduces the model confidence lower than the model confidence when all features are perturbed.

2.2.5 SensX landscape reveals a minimal set of only 1.5% of the total features that precisely demarcate the facial attributes that the smiling-ViT has learned as important.

We define the minimal set of important SensX features as the smallest k such that perturbing top k SensX features reduces the model confidence below 50%. SensX landscape for smiling-ViT reveals that perturbing only 2271 features

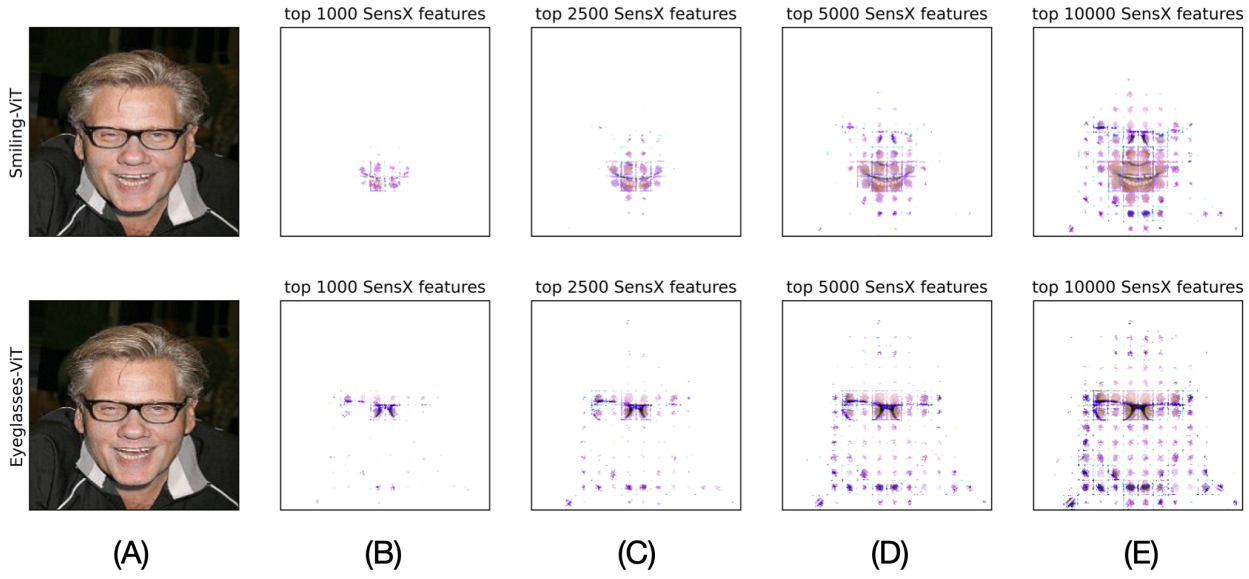


Figure 4: (A) Test image. (B)-(E) Images with top $k = 1000, 2500, 5000, 10000$ SensX features for smiling-ViT (top panel) and eyeglasses-ViT (bottom panel) having their corresponding value in the test image and the remaining features set to 0.

out of the total of $150,528$ features reduces the model confidence below 50%. Figure 5B (top panel) shows that the mask defined by these features demarcates the smile in the human face in the test image. On the other hand, perturbing at least top $39,335$ SensX features (26% of total features) reduces the median confidence of eyeglasses-ViT to below 50%. Figure 5B (bottom panel) shows the mask of these features in the test image.

2.2.6 Instance-wise analysis shows that important features revealed by SensX are input-specific and accurate.

Figure 6 shows the importance of input-wise XAI. SensX values show that the ViTs have correctly learned facial attributes specific to different human faces.

2.2.7 SensX at full resolution reveals possible biases inherent to the model architecture.

We notice that the heat maps of SensX values shown in Figure 3 reveal a grid-like structure—features on the grid boundaries and at the center of the grid patches are brighter than their neighbors. This grid is of size 16×16 that matches exactly with the patch-size of tokens in the ViT architecture [Dosovitskiy, 2020]. This suggests that there may be a bias towards the edges and center of the input tokens during training of this ViT architecture.

2.3 Automated cell type annotation using single-cell RNA data set

Single-cell RNA sequencing (scRNA-seq) quantifies gene expression levels in individual biological cells by measuring abundance of mRNA molecules per cell [Heumos et al., 2023]. Advancements in experimental technology have resulted in large data sets with tens of thousands of genes measured in millions of individual cells. Annotating cell types using scRNA-seq data is a fundamental problem [Hou and Ji, 2024]. Various automated methods for cell annotation using different AI architectures have been proposed, for example, transformers [Chen et al., 2023], graph neural networks [Shao et al., 2021], and FNNs [Ma and Pellegrini, 2020]. For biological validation and hypotheses development, it is necessary to explain what black-box AI models trained for automated cell annotation have learned as distinguishing genes for different cell types.

We implemented FNNs to learn classification of cell types using the publicly available scRNA-seq data set of the human lung cell atlas (HLCA) [Sikkema et al., 2023]. The data set has $1,305,099$ cells with gene expression values reported for $56,239$ genes per cell. We trained binary classifiers to identify 21 different cell types (see Figure7B for a list of the cell types).

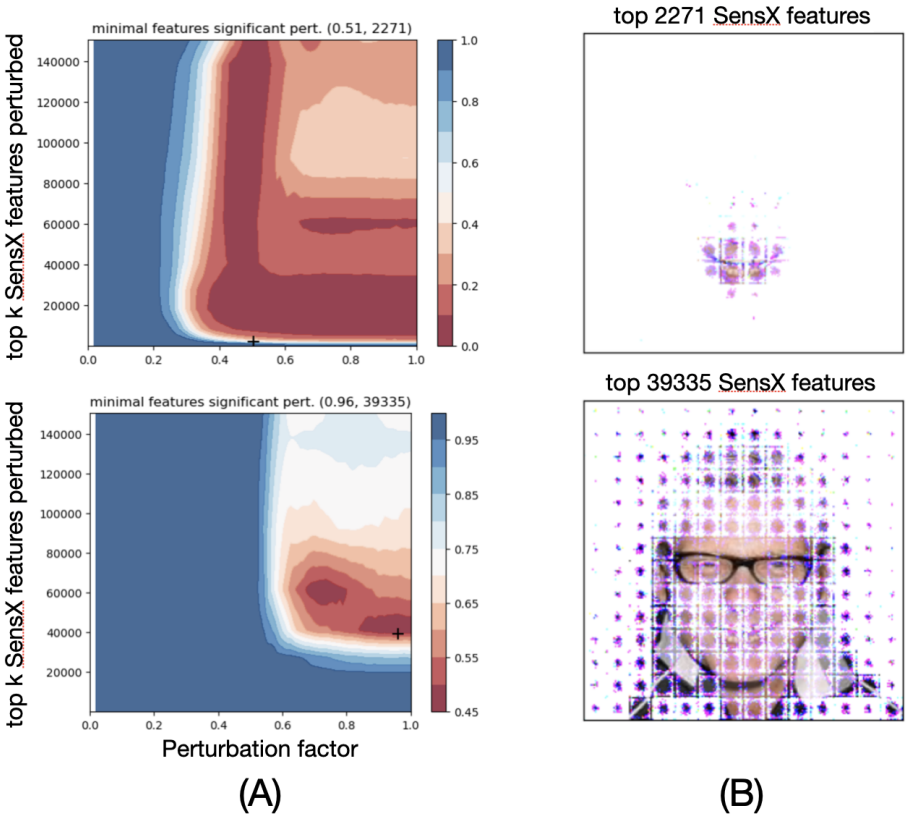


Figure 5: (A) Heatmaps of median model confidence when different number of top SensX features are perturbed using different values of perturbation factors for smiling-ViT (top) and eyeglasses-ViT (bottom). '+' marks a minimal set of features to be perturbed to reduce the model confidence below 50%. (B) Masks of the image defined by the minimal set of top SensX features for smiling-Vit (top) and eyeglasses-ViT (bottom).

2.3.1 SensX determines an optimal set of important genes for each cell type.

To compute an optimal set of important genes, we consider both the number of top k genes and δ. This is because a higher value of δ might result in a local neighborhood of the input that spans multiple log scales (gene expression values in scRNA-seq data are log-normalized). Hence, it may be of biological interest to minimize δ as well to avoid perturbations that imply changing gene expression values too widely. Hence, we define the Pareto optimal as the pair (k, δ) with minimal norm that reduces the median model confidence below 50%. Figure 7A shows SensX landscape of a single cell identified by the mast model with high confidence. The Pareto optimal point is marked by an 'x'. Figure 7B shows the median and CQD values of the Pareto optimal points computed for 1000 cells of each cell type. For all but one cell type, perturbing 284 top SensX genes (0.5% of the total genes) by a perturbation factor of 0.1 to 0.25 reduces the median model confidence below 50%.

2.3.2 SensX combines multiple instance-wise analyses to compare features important for populations of cells

We computed SensX for 1000 cells of each cell type from the test data. Cells belonging to different cell types do not cluster distinctly in a three-dimensional embedding of gene expression values (see Figure 8A). However, the sets of optimal genes determined by SensX for each cell, cluster distinctly in a three dimensional embedding (see Figure 8B).

3 Discussion

The underlying assumption in the framework is that uniformly distributed perturbations significantly affect the QOI. Therefore, the QOI should be chosen judiciously. SensX provides a criterion to justify the choice of QOI—$\delta^* > 0$ (see Methods) . The QOI of the case studies was chosen as the confidence of the model that the given input belongs to the class. It is generally assumed that uniform perturbations represent noise and that data representing a specific class is

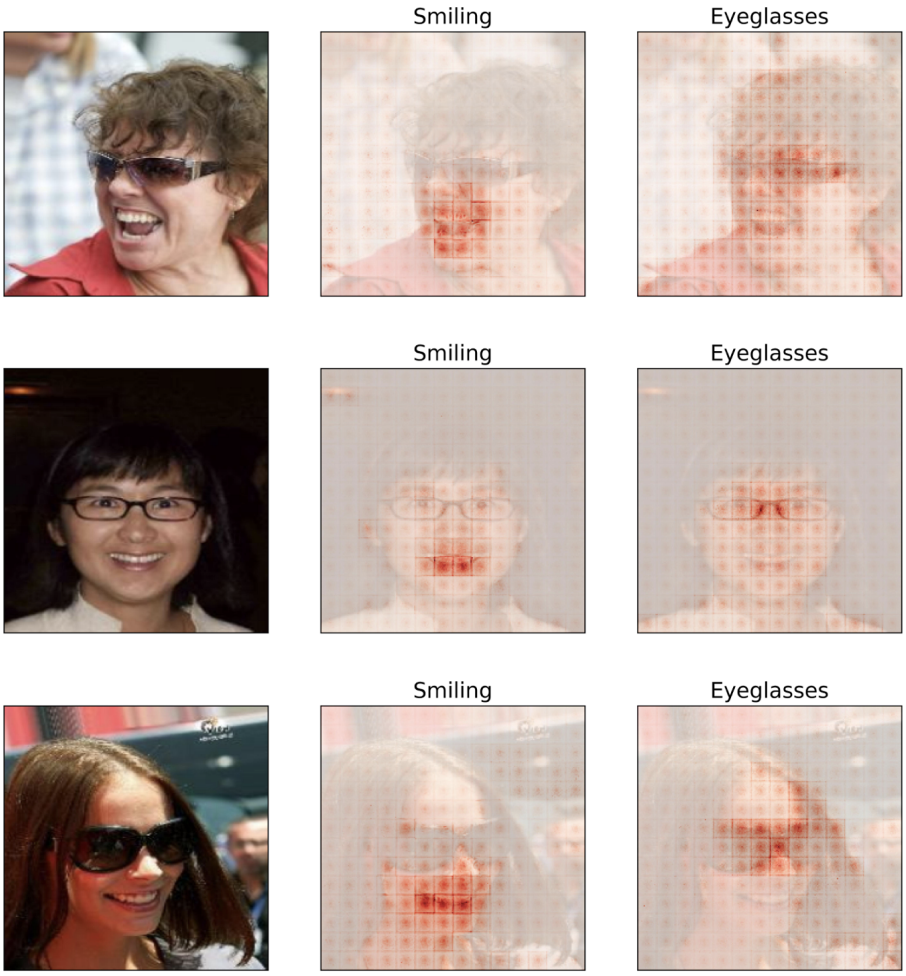
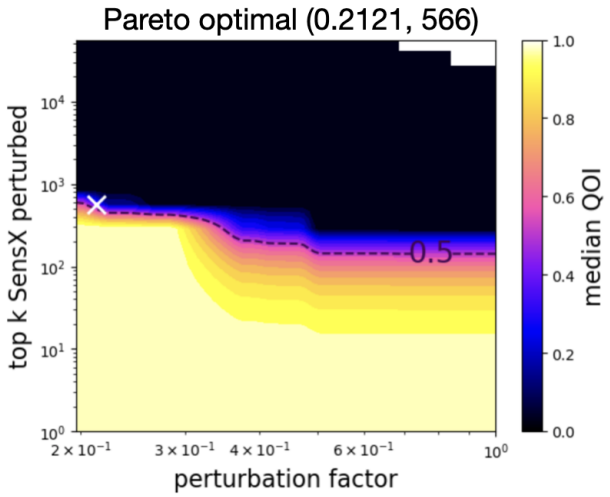


Figure 6: SensX shows that ViT models have learned input-specific features accurately.

distinguished by a specific pattern or relations in the features that are not uniformly distributed. However, this may not be the case always (see supplementary section for fashion MNIST example).

The number of trajectories is an important hyperparameter of SensX to balance the convergence of GSA measures and computation time. There is no theoretical basis to compute an optimal value, but we provide the following strategy to make a judicious choice. A computational advantage of SensX is that it can be computed in batches without any additional overhead, and the results of any number of batches can be combined. Hence, it is advisable to perform computations in batches, and the convergence of GSA measures can be assessed by combining all batches. We propose a possible criterion for convergence of GSA measures in the supplementary.

We note that all perturbations are uniform and independent within the justifiable domain. We do not assume any correlations between the features a priori. The aim of SensX is to determine the correlations between features that a trained model has learned, and it does so by GSA using independent perturbations to explore the input space. Given the trained model, the optimal set of top SensX features corresponds to an optimal set of correlated features that may be further analyzed to determine precise signed correlations.

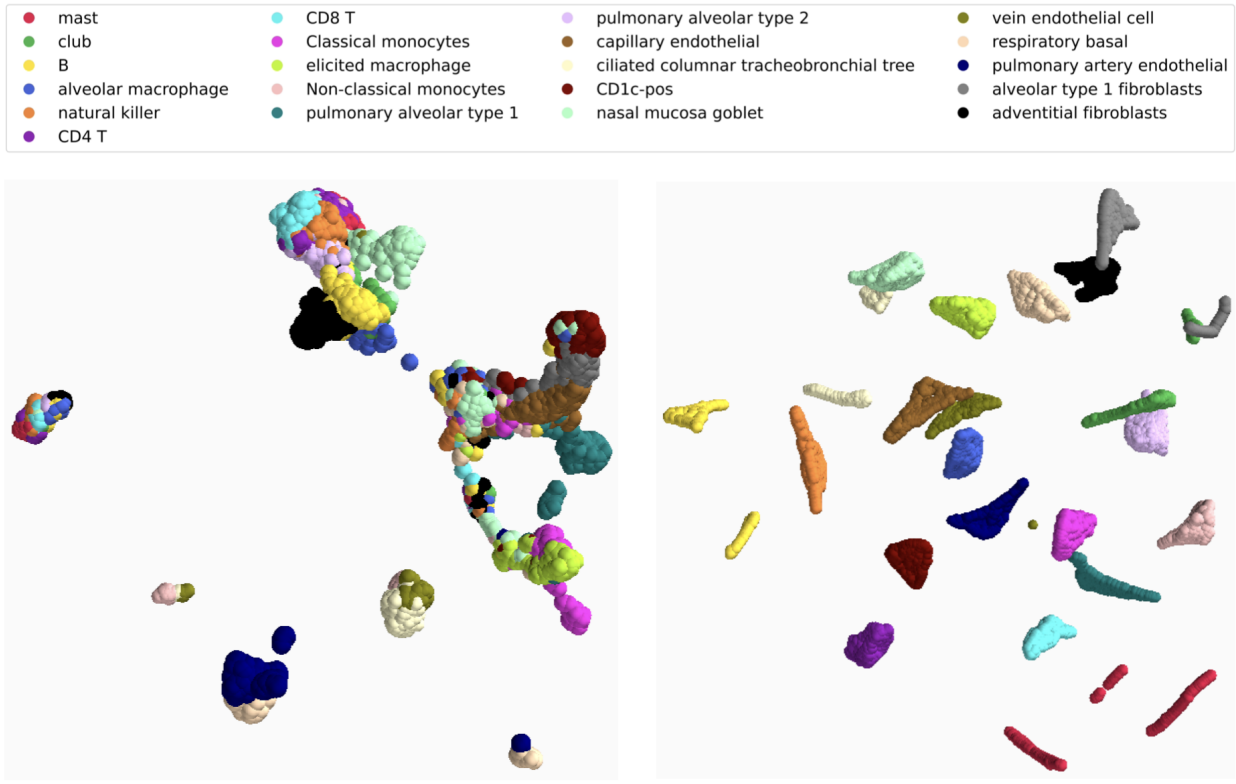


(A)

cell type	perturbation factor		top k SensX perturbed	
	median	CQD	median	CQD
mast	0.24	0.14	566	0.33
club	0.09	0.18	284	0.00
B	0.25	0.14	284	0.33
alveolar macrophage	0.14	0.17	284	0.00
natural killer	0.14	0.24	284	0.00
CD4 T	0.09	0.26	284	0.00
CD8 T	0.12	0.17	284	0.00
Classical monocytes	0.13	0.20	284	0.00
elicited macrophage	0.08	0.25	284	0.00
Non-classical monocytes	0.13	0.25	284	0.00
pulmonary alveolar type 1	0.21	0.25	284	0.00
pulmonary alveolar type 2	0.20	0.17	284	0.00
capillary endothelial	0.17	0.21	284	0.00
ciliated columnar tracheobronchial tree	0.28	0.14	284	0.33
CD1c-pos	0.10	0.30	284	0.00
nasal mucosa goblet	0.13	0.20	284	0.00
vein endothelial cell	0.23	0.17	284	0.00
respiratory basal	0.14	0.26	284	0.00
pulmonary artery endothelial	0.15	0.23	284	0.00
alveolar type 1 fibroblasts	0.25	0.19	284	0.00
adventitial fibroblasts	0.20	0.21	284	0.00

(B)

Figure 7: (A) SensX landscape of a single mast cell. 'x' marks the Pareto optimal that minimizes the number of top SensX features to be perturbed and the perturbation factor. (B) Median and CQD of the Pareto optimal perturbation factor and the number of SensX features to be perturbed for 1000 cells of each of the different cell types.



(A) (B)

Figure 8: (A) UMAP [McInnes et al., 2018] 3D-embedding of the gene expression values of 1000 cells of each cell type do not show distinct clusters for the different cell types. (B) UMAP embedding of the Pareto optimal SensX features show distinct clusters for each cell type.

4 Methods

4.1 Framework of SensX

4.1.1 Terminology

We consider deep learning models that are trained to learn binary classification.

A labeled data set of N samples with K features each is a set of tuples $\mathcal{D} = \{(\mathbf{x}_i, b_i)\}_{1 \leq i \leq N}$, where $\mathbf{x}_i \in \mathbb{R}^K$ and $b_i \in \{0, 1\}$. The deep learning model is a continuous function parameterized by M parameters that takes a data sample as input and outputs a pair of real numbers. We denote the model by $m(\mathbf{x}; \phi) = \mathbf{y}$ where $\phi \in \mathbb{R}^M$ and $\mathbf{y} \in \mathbb{R}^2$. A classifier function $q(\mathbf{y}) \equiv q(m(\mathbf{x}; \phi)) \equiv q(\mathbf{x}) = \mathbf{p}$ normalizes $\mathbf{y}$ to compute a tuple $\mathbf{p} \in [0, 1]^2$ corresponding to the probabilities of the two labels predicted by the model for the given $\mathbf{x}$.

4.1.2 Meaningful perturbations: Global domain and local neighborhoods

Let $m(\mathbf{x}_i; \phi^*) \equiv m_\phi(\mathbf{x})$ be the trained deep learning model to be analyzed. We define the global domain as the Cartesian product of constraints on all features, denoted by $\Omega^g = \prod_{1 \leq j \leq K} [f_j^-, f_j^+]$ where f_j^- and f_j^+ are lower and upper bounds of the j-th feature. In each application shown in this work, $f_j^- = \min_{i \in A}\{x_{ij}\}$ and $f_j^+ = \max_{i \in A}\{x_{ij}\}$. In other words, Ω^g was chosen as the smallest hypercube containing all training set samples. However, the framework accepts any user-defined Ω^g that may be contextually relevant based on expert advice.

We define a local neighborhood of $\mathbf{x} = (x_1, ..., x_K) \in \Omega^g$ parameterized by $\delta \in [0, 1]$ as

$$\Omega(\mathbf{x}, \delta) = \prod_{1 \leq j \leq K} [\max\{f_j^-, x_j - \delta r_j\}, \min\{f_j^+, x_j + \delta r_j\}]. \tag{1}$$

δ is called the perturbation factor. It follows from equation 1 that $\Omega(\mathbf{x}, \delta) \subseteq \Omega^g$ and $\mathbf{x} \in \Omega(\mathbf{x}, \delta)$.

4.1.3 Significant perturbations: Optimal local neighborhood for perturbations

Given $\mathbf{x} \in \Omega^g$, we define $T_{\mathbf{x},\delta,n_o}$ as the set of n_o randomly generated points in $\Omega(\mathbf{x}, \delta)$. n_o and $\mathbf{x}$ are fixed for the purposes of this discussion and $\delta \in \Delta$, a finite set of values for the perturbation factor. Hence, we will drop $\mathbf{x}$ and n_o for notational convenience. We say,

$$T_\delta = \{\tilde{\mathbf{x}} | n_o \text{ points uniformly selected from } \Omega(\mathbf{x}, \delta)\}. \tag{2}$$

We define

$$\widehat{q}(\delta) = \text{median}(\{q(\tilde{\mathbf{x}}) | \tilde{\mathbf{x}} \in T_\delta\}). \tag{3}$$

We define optimal perturbation factor δ^* as the maximal value in Δ such that $\left|\frac{d\widehat{q}}{d\delta}\right| < \tau_s \forall \delta \geq \delta^*$, where $\tau_s > 0$ is a user-defined threshold.

By default, $n_o = 1000, \Delta = \{0.02m | m \in \mathbb{Z}_{50}\}$, and $\tau_s = 0.1$. $\frac{d\widehat{q}}{d\delta}$ is estimated using `numpy.gradient`.

4.1.4 Grounded perturbations: Modified Morris method to compute SensX values

Let $E = \{\tilde{\mathbf{e}} | n_w$ samples uniformly selected from $\Omega(\mathbf{x}, \delta^*_{\mathbf{x}})\}$ where n_w denotes the number of walks, a user-defined hyperparameter. We define a trajectory in $\Omega(\mathbf{x}, \delta^*_{\mathbf{x}})\}$ starting at $\mathbf{x}$ and ending at $\tilde{\mathbf{e}} \in E$ as a tuple of points in $\Omega(\mathbf{x}, \delta^*_{\mathbf{x}})\}$, denoted by

$$T(\tilde{\mathbf{e}}) = (\mathbf{t}_{i_0} = \mathbf{x}, \mathbf{t}_{i_1} ..., \mathbf{t}_{i_K}), \tag{4}$$

where $(i_1, ..., i_K)$ is a randomly permuted sequence of integers from 1 to K, and for $k > 0$,

$$\mathbf{t}_{i_k} = (t_{i_k j}) = \begin{cases} e_j \text{ if } j \in \{i_1, i_2, ..., i_k\} \\ x_j, \text{ otherwise.} \end{cases} \tag{5}$$

The elementary effect of feature i_k along the trajectory T is defined by

$$s_{i_k}^{T(\tilde{\mathbf{e}})} = \left| \frac{g(\mathbf{t}_{i_k}) - g(\mathbf{t}_{i_{k-1}})}{e_{i_k} - x_{i_k}} \right|. \tag{6}$$

We define the SensX value of feature i_k as the average of elementary effects over all trajectories,

$$\widehat{s}_{i_k} = \frac{1}{n_w} \sum_{\tilde{\mathbf{e}} \in E} s_{i_k}^{T(\tilde{\mathbf{e}})}. \tag{7}$$

This formulation differs from the traditional Morris method in that the starting point of all trajectories is the same, specifically, it is the model input being analyzed. This is an important distinction because in this context, it is of interest to compute sensitivities with reference to the model input under consideration.

4.2 Computational implementation

We implemented SensX in Python using the Google Just After eXecution (JAX) machine learning library [Bradbury et al., 2018]. For computational efficiency, we designed our algorithm using pure functions and without side-effects to allow just-in-time (JIT) compilation. We implemented the deep learning models in this work using NNX, a JAX-based neural network library. Supplementary file `conda_packages.txt` shows versions of all packages in the conda environment to reproduce the results. All model simulations were done on NVIDIA A100 GPUs.

For efficient parallelization, two different algorithms to compute SensX are provided—batching over inputs or data samples to be analyzed and batching over the features per data sample. The latter is recommended if the number of features is much larger than the number of samples and the model simulation is expensive. Since a trajectory has an inherent order, batching over features using pure functions that allow JIT compilation is non-trivial (see Supplementary for details).

4.3 Synthetic data sets

4.3.1 Generating labeled data

Samples are generated from ten-dimensional standard Gaussian distribution. Each sample, $\mathbf{x} = (x_1, ..., x_{10})$ is labeled $b \in \{0, 1\}$ based on the following criteria for three of the four synthetic data sets (`https://github.com/Jianbo-Lab/CCM` >).

- XOR model: $P(b = 1) \propto x_1 x_2$
- orange skin model: $P(b = 1) \propto \sum_{1 \le i \le 4} x_i^2 - 4$
- nonlinear additive model: $P(b = 1) \propto -100 * \sin 0.2x_1 + |x_2| + x_3 + e^{-x_4} - 2.4$

The switch data set is a mixture of data samples from two different criteria as follows. x_{10} is generated from a mixture of two Gaussians centered at ± 3 with equal probability. If x_{10} is generated from the Gaussian centered at 3, then x_1 to x_4 are used to label the sample based on the orange skin model. Otherwise, x_5 to x_8 are used to label the sample based on the nonlinear additive model.

4.3.2 Model and training

We generated $100,000$ labeled data samples for each data set and used FNNs with layers $[10, 200, 100, 50, 2]$ to learn the classifications. We used 80% of the samples as the training set and the rest as the test set. Models were trained by minimizing the crossentropy loss using the AdamW method with the constant learning rate of 0.0001. Each optimization step consisted of a batch of 100 randomly-picked samples from the training set. We implemented $10,000$ optimization steps and computed the F1-score for the test set after every 100 steps. The models analyzed using SHAP and SensX were defined by the sets of parameters that yielded the maximal F1-score during training <Table in supplementary with F1-scores>.

4.4 Vision transformer (ViT)

4.4.1 Model setup

We implemented the vision transformer model defined at `https://docs.jaxstack.ai/en/latest/JAX_Vision_transformer.html` and loaded the model parameters pre-trained on the ImageNet-21k data set from the transformers Python library [Wolf et al., 2020]. We replaced the model classifier with a fully connected layer with two outputs.

To compute smiling-ViT, we fine-tuned the model by training it on the publicly available images of human faces from the Large-scale CelebFaces Attributes (CelebA) data set [Liu et al., 2015] to predict label 0 for faces annotated not smiling and to predict label 1 for those annotated smiling. Similarly, we computed eyeglasses-ViT to predict eyeglasses on image of a human face.

Acknowledgements

This research was supported by the Intramural Research Program of the NIH, the National Institute of Diabetes and Digestive and Kidney Diseases (NIDDK), project number Z01-DK075146. This work utilized the computational resources of the NIH HPC Biowulf cluster (https://hpc.nih.gov).

References

Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

Alex J DeGrave, Joseph D Janizek, and Su-In Lee. Ai for radiographic covid-19 detection selects shortcuts over signal. *Nature Machine Intelligence*, 3(7):610–619, 2021.

David Gunning and David Aha. Darpa's explainable artificial intelligence (xai) program. *AI magazine*, 40(2):44–58, 2019.

Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information fusion*, 58:82–115, 2020.

Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. Explainable ai: A review of machine learning interpretability methods. *Entropy*, 23(1):18, 2020.

Zachary C Lipton. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57, 2018.

Ilya M Sobol. Global sensitivity indices for nonlinear mathematical models and their monte carlo estimates. *Mathematics and computers in simulation*, 55(1-3):271–280, 2001.

Michel Tosin, Adriano MA Côrtes, and Americo Cunha. A tutorial on sobol'global sensitivity analysis applied to biological models. *Networks in Systems Biology: Applications for Disease Modeling*, pages 93–118, 2020.

Timo Ikonen. Comparison of global sensitivity analysis methods–application to fuel behavior modeling. *Nuclear Engineering and Design*, 297:72–80, 2016.

Bas Van Stein, Elena Raponi, Zahra Sadeghi, Niek Bouman, Roeland CHJ Van Ham, and Thomas Bäck. A comparison of global sensitivity analysis methods for explainable ai with an application in genomic prediction. *IEEE Access*, 10: 103364–103381, 2022.

Livia Paleari, Ermes Movedi, Michele Zoli, Andrea Burato, Irene Cecconi, Jabir Errahouly, Eleonora Pecollo, Carla Sorvillo, and Roberto Confalonieri. Sensitivity analysis using morris: Just screening or an effective ranking method? *Ecological Modelling*, 455:109648, 2021.

Francesca Campolongo, Jessica Cariboni, and Andrea Saltelli. An effective screening design for sensitivity analysis of large models. *Environmental modelling & software*, 22(10):1509–1518, 2007.

Scott Lundberg. A unified approach to interpreting model predictions. *arXiv preprint arXiv:1705.07874*, 2017.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International conference on machine learning*, pages 3145–3153. PMlR, 2017.

Jianbo Chen, Le Song, Martin Wainwright, and Michael Jordan. Learning to explain: An information-theoretic perspective on model interpretation. In *International conference on machine learning*, pages 883–892. PMLR, 2018.

Shams Mehdi and Pratyush Tiwary. Thermodynamics-inspired explanations of artificial intelligence. *Nature Communications*, 15(1):7859, 2024.

Jianbo Chen, Mitchell Stern, Martin J Wainwright, and Michael I Jordan. Kernel feature selection via conditional covariance minimization. *Advances in neural information processing systems*, 30, 2017.

Zoltán Botta-Dukát. Quartile coefficient of variation is more robust than cv for traits calculated as a ratio. *Scientific Reports*, 13(1):4671, 2023.

Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

Lukas Heumos, Anna C Schaar, Christopher Lance, Anastasia Litinetskaya, Felix Drost, Luke Zappia, Malte D Lücken, Daniel C Strobl, Juan Henao, Fabiola Curion, et al. Best practices for single-cell analysis across modalities. *Nature Reviews Genetics*, 24(8):550–572, 2023.

Wenpin Hou and Zhicheng Ji. Assessing gpt-4 for cell type annotation in single-cell rna-seq analysis. *Nature Methods*, pages 1–4, 2024.

Jiawei Chen, Hao Xu, Wanyu Tao, Zhaoxiong Chen, Yuxuan Zhao, and Jing-Dong J Han. Transformer for one stop interpretable cell type annotation. *Nature Communications*, 14(1):223, 2023.

Xin Shao, Haihong Yang, Xiang Zhuang, Jie Liao, Penghui Yang, Junyun Cheng, Xiaoyan Lu, Huajun Chen, and Xiaohui Fan. scdeepsort: a pre-trained cell-type annotation method for single-cell transcriptomics using deep learning with a weighted graph neural network. *Nucleic acids research*, 49(21):e122–e122, 2021.

Feiyang Ma and Matteo Pellegrini. Actinn: automated identification of cell types in single cell rna sequencing. *Bioinformatics*, 36(2):533–538, 2020.

Lisa Sikkema, Ciro Ramírez-Suástegui, Daniel C Strobl, Tessa E Gillett, Luke Zappia, Elo Madissoon, Nikolay S Markov, Laure-Emmanuelle Zaragosi, Yuge Ji, Meshal Ansari, et al. An integrated cell atlas of the lung in health and disease. *Nature medicine*, 29(6):1563–1577, 2023.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/jax-ml/jax`.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/2020.emnlp-demos.6`.