

# Deep Neural Networks Interpretation and Applications for Ultrasound and Cell Profiling

Thesis submitted in partial fulfillment of the requirements for the degree of

"DOCTOR OF PHILOSOPHY"

by

**Alon Shpigler**

Submitted to the Senate of Ben-Gurion University of the Negev

*7<sup>th</sup> May, 2024*

Beer-Sheva

# Deep Neural Networks Interpretation and Applications for Ultrasound and Cell Profiling

Thesis submitted in partial fulfillment of the requirements for the degree of

"DOCTOR OF PHILOSOPHY"

by

**Alon Shpigler**

Submitted to the Senate of Ben-Gurion University of the Negev

Approved by the Advisors

Approved by the Dean of the Kreitman Scholar of Advanced Graduate Studies

*7<sup>th</sup> May, 2024*

Beer-Sheva



This work was carried out under the supervision of Prof. Assaf Zaritsky and Prof. Hillel Bar-Gera in the Department of Industrial Engineering and Management, Faculty of Engineering Science, Ben-Gurion University of the Negev

**Alon Shpigler**

*Deep Neural Networks Interpretation and Applications for Ultrasound and Cell Profiling*

7<sup>th</sup> May, 2024

Supervisors: Assaf Zaritsky & Hillel Bar-Gera

**Ben-Gurion University of the Negev**

**Faculty of Engineering Sciences**

Department of Industrial Engineering and Management

Marcus Family Campus, Ben-Gurion University of the Negev, Israel

P.O.B. 653 8410501 Be'er Sheva

# Declaration


I, Alon Shpigler, whose signature appears below hereby declare that:

- I have written this thesis by myself, except for the help and guidance offered by my thesis advisors.
- The scientific materials included in this thesis are products of my own research, culled from the period during which I was a research student.

Date: 7<sup>th</sup> May, 2024,

Student's name: Alon Shpigler

Signature:



---

# Acknowledgments

I would like to express my gratitude to everyone who has supported me throughout my Ph.D. journey.

Firstly, I would like to thank my mentors, Prof. Assaf Zaritsky and Prof. Hillel Bar-gera, as well as my past Ph.D. mentors, Dr. Aharon Bar-Hillel and Dr. Etai Mor, for their insightful comments, constructive feedback, and patience which have been invaluable to my research. I have learned profound lessons from all of them, which have helped me become a better researcher and human being. I would like to especially thank Prof. Assaf Zaritsky, for his unwavering support, guidance, mentorship, and commitment to my development and success.

I would also like to extend my appreciation to the members of my thesis escort committee for their invaluable feedback and suggestions, which have significantly contributed to the quality of my work.

I am deeply grateful to the IEM and SISE departments for providing me with access to the necessary resources and facilities, including computing resources and research materials, which have been instrumental in the completion of my research.

I want to thank my lab colleagues and friends for their encouragement, motivation, and support throughout my Ph.D. years. Their valuable insights, discussions, and collaborations have played an essential role in shaping my research ideas.

Lastly, I want to express my heartfelt gratitude to my friends and family for their unwavering love, support, and encouragement throughout my academic journey. Their patience, sacrifice, and understanding have been instrumental in enabling me to achieve this significant milestone in my life. I would like to give special thanks to my partner, Yaeli, who has been there to tolerate me throughout the ups and downs over the last few years and has made the journey possible for me, and much nicer.

My Ph.D. journey wouldn't have been the same without each of you.

# Contents

<b>1</b>	<b>Dissertation in a Nutshell</b>	<b>15</b>
<b>2</b>	<b>SIGN: Statistical Inference Graphs Based on Probabilistic Network Activity Interpretation</b>	<b>18</b>
2.1	Introduction . . . . .	19
2.2	Related Work . . . . .	22
2.3	Method . . . . .	24
2.4	Results . . . . .	35
2.5	Conclusions . . . . .	46
<b>3</b>	<b>Detection of Overlapping Ultrasonic Echoes with Deep Neural Networks</b>	<b>48</b>
3.1	Introduction . . . . .	48
3.2	Problem Formulation and Related Work . . . . .	51
3.3	Echo Detection using Deep Networks . . . . .	52
3.4	Experimental Setup . . . . .	56
3.5	Results . . . . .	57
3.6	Conclusions . . . . .	63
<b>4</b>	<b>Anomaly Detection for High-Content Image-Based Phenotypic Cell Profiling</b>	<b>64</b>
4.1	Introduction . . . . .	64
4.2	Results . . . . .	66
4.3	Anomaly-based Representations are Interpretable . . . . .	73
4.4	Discussion . . . . .	75
4.5	Methods . . . . .	77
4.6	Supplementary Information . . . . .	80
<b>5</b>	<b>Final Remarks</b>	<b>84</b>
	<b>Bibliography</b>	<b>85</b>

# List of Figures

2.1	<p><b>Explaining network errors with SIGN.</b> An ILSVRC-2012 [13] dataset image of a pineapple (red-circled), falsely classified by VGG-16 to the “swing” class. This is a partial inference graph with “swing” as the inferred class at the top, and the 3 most influential visual words from a high level layer (block5_conv1 layer), forming it. The 3 visual words show <i>what</i> the network found to be the most important features for classification of the inspected image. Each such word is represented using 6 images containing it, with the visual word itself in a red rectangle. The analyzed image is presented above each visual word, with red dots showing <i>where</i> this visual word is found. As seen in the graph, the image is classified based on 3 “swing”-related visual words describing “rope”, “sand” and “grass”. The full inference graph is seen in Fig. 2.9 Section 2.4.3. . . . . .</p>	20
2.2	<p><b>Illustration of SIGN framework flow.</b> (A) Modeling neural network layers as arising from a probabilistic generative model, and forming a visual word dictionary to each modeled layer. Training the generative model can be simultaneously with the network or post-hoc. (B) Forward pass through the modeled network on a subset inputted test images. Calculate the co-occurrence matrix statistics for all visual words that appear in the image. Apply Node Selection Algorithm on the matrix to obtain the most explanatory visual words. Produce an image inference graph of words as nodes and their connection strengths as edges. (C) The same as (B) but with a subset of analyzed images that represent a class predicted by the network. . . . . .</p>	21
2.3	<p><b>A graphical model for MLP networks.</b> Each orange rectangle is a layer activation vector after the ReLU operation. Activation <math>x^l[d]</math> of neuron <math>d</math> in layer <math>l</math> is assumed to be generated from a rectified Gaussian density, resetting values lower than zero to zero. <math>y^l[d]</math> is the parent of <math>x^l[d]</math>, describing the original Gaussian density before it was rectified. <math>h^l</math> is a hidden variable generating the hidden vector of multivariate Gaussians <math>Y^l</math>. <math>h^l</math> takes <math>K^l</math> different states, creating a mixture of multivariate Gaussians for layer <math>l</math>. . . . . .</p>	24
2.4	<p><b>Two inference paths in MLP.</b> We show the main decision junctions for an example six-layer MLP network. The analyzed examples are presented on the left. Cluster representatives are chosen using the <math>l_2</math> metric. In each decision junction, the points of the sub-cluster chosen by the example are marked with full circles in cyan (top) blue (bottom) <b>Top:</b> The path of a misclassified CIFAR10 car example is shown. The main decision points occurred in layers fc-2 – fc-4, with the critical decision made in layer fc-3 where the abnormal appearance of the example misled the network to consider the car as a truck. <b>Bottom:</b> The path of a misclassified MNist “9” digit example is shown. The input image traversing main decision clusters through layers fc-1 – fc-4, where the main flawed decision is made in layer fc-3, where the open head of the “9” image misleads the network to consider as “4” digit. . . . . .</p>	34

2.5	<b>Cluster development across layers.</b>	Cluster similarity matrices for increasing layer indices in MLP and CNN networks trained for CIFAR10 classification. In each matrix, Euclidean distances between cluster centers for clusters of a single layer are shown. Clusters are ordered by their dominant class (clusters with dominant class 1, followed by all clusters with dominant class 2, etc.). Layer index and average cluster purity (%) are shown above each matrix. <b>Top: MLP activity.</b> Growing similarity between clusters with the same dominant class can be seen by the appearance of a block diagonal structure, from layer 3 onwards. This indicates convergence toward a single class-specific representation as layers progress. <b>Bottom: CNN activity.</b> Clusters of add-layers of ResNet blocks 1, 3, 5, 7 and 8 are presented. Here, no similarity is formed with layer growth; each class is represented by multiple localized visual words, which have unrelated activity patterns. . . . .	36
2.6	<b>Cluster development for an extreme overfit case.</b>	Cluster distances and average purity for a network trained with random labels are shown. <b>Top:</b> Cluster similarity matrices for layers 1-5. <b>Bottom:</b> Typical clusters at these layers. For each cluster the (pseudo) label histogram is shown, as well as some representative cluster images. An abrupt transition from input-dominated to output-dominated representation occurs in the transformation between the third and fourth layers. . . . .	37
2.7	<b>Classification accuracy analysis as a function of dictionary size and layer depth.</b>	Error rates of linear classifiers, based on cluster/visual word histograms are shown. All experiments conducted on five ResNet20 conv-layers, trained on CIFAR10. <b>Left:</b> Error rates based on SIGN generative and discriminative loss functions (Eqs. 2.16 and 2.17), as a function of dictionary size. <b>Right:</b> Error rates of SIGN clustering method and plain clustering based on the maximal active channel (see text for explanation). . . . .	38
2.8	<b>Pineapple inference graph.</b>	The graph is generated by training a model on the "pineapple" class and its neighboring classes. The top node is a visual word of the output layer, representing the predicted class "pineapple". The lower levels in the graph show the three most influential words in preceding modeled layers (block5_conv1, ..., block1_conv1). Visual words are manifested by the six representative examples for which $P(h^l = k x_p^l)$ is the highest. For modeled layer block5_conv1, examples are presented by showing the example image with a rectangle highlighting the receptive field of the word's location. For lower layers, the receptive field patches themselves are shown. Images are annotated by their true label. Arrows are shown for the two most significant connections for each lower visual word. When the log-ratio term (right element in Eq. 2.25) is positive, it is colored (1) black: $0 < \text{log-ratio} < 1$ , (2) light green: $1 \leq \text{log-ratio} < 2$ , (3) mild green: $2 \leq \text{log-ratio} < 3$ , or (4) dark green: $3 \leq \text{log-ratio}$ . In addition, a tag above each visual word was added by the authors for convenience. The figure is best inspected by zooming in on clusters of interest. . . . .	39
2.9	<b>An image inference graph of an erroneous image.</b>	An image inference graph for a pineapple image wrongly classified to the class "swing". The model is trained using "pineapple" and its neighboring classes (same as in Fig. 2.8), where the neighbor class "swing" is included. The graph is generated by applying the node selection algorithm (Section 2.3.3) to a set $\Omega$ containing this single erroneous image. The analyzed image is shown on the top of each cluster node, with red dots marking spatial locations assigned to the cluster. In the top node, the pineapple object is marked in red circle for clarification. . . . .	40
2.10	<b>Partial image inference graph of a correctly classified zebra image.</b>	The figure presents only the main contributing visual words from the three bottom modelled layers. . . . .	42

2.11	<b>Sub-graph inference for ResNet50.</b> <b>Left:</b> The pineapple image wrongly classified to class "swing" in VGG-16, presented in Fig. 2.9, is correctly classified to "pineapple" class in ResNet50 (the pineapple object is marked in green circle). The sub-graph presents the visual word from the top layer (add_16) connected to a visual word from the lower layer (add_13). <b>Right:</b> The zebra image from Fig. 2.10, correctly classified in VGG-16, is correctly classified in ResNet50 as well. The sub-graph includes a visual word from the top layer (add_16) aggregated from two visual words from the lower layer (add_13). . . . .	43
2.12	<b>Biased data experiment results.</b> Users were asked to determine whether inference graphs shown to them contain corrupted data. The graphs shown were produced using ResNet20 trained either with clean data (blue), or with opaque/transparent watermarks induced in 2 classes of CIFAR10. For the datasets with watermarks, users were shown inference graphs of corrupted classes (green) and non-corrupted classes (orange). Users easily detected corruptions in class inference graph of classes stained with watermarks. When shown inference graphs of non-corrupted classes from the same corrupted dataset, the user accuracy is lower, especially with transparent watermarks. . . . .	44
2.13	<b>Corrupted data debugging with SIGN.</b> Class inference graph of "truck" class from ResNet20, trained on biased CIFAR10. The data was corrupted by inducing 2 classes with transparent watermarks ( $\alpha = 0.5$ ) at random places. The above graph shows the "truck" class inference graph in an experiment where classes "truck" and "cat" were corrupted with purple letters "T" and "A" respectively. It is recommended to zoom in for better inspection. . . . .	45
2.14	<b>Attention maps demonstration on biased data.</b> (a) Examples of CIFAR10 images corrupted with transparent watermarks along with their (b) corresponding Grad-CAM [9] heatmaps. . . . .	46
3.1	Illustrations of ultrasonic overlapping echoes (top) along with their envelopes and TOFs (bottom). (a) Minor overlap, (b) major overlap, (c) successive overlaps, and (d) overlap with a low-amplitude echo. . . . .	49
3.2	<b>Simulated data illustration.</b> Examples of Simulated signals (top) used for network training and testing, along with their envelopes and TOFs (bottom). . . . .	54
3.3	<b>Schematic overview of the suggested US-CNN architecture.</b> . . . . .	55
3.4	<b>Representative filters learned by the first layer of the network.</b> <b>Left:</b> Filter of a network trained with one filter in the first layer. <b>Right:</b> 3 Representative filters of a network trained with 30 filters in the first layer. . . . .	56
3.5	<b>Simulation results.</b> Comparison of detection accuracy as measured by AUC (top) and F1 (bottom) among the four methods tested. Detection accuracy is plotted as a function of echo overlap parameter $\lambda$ (right), required accuracy threshold $\epsilon$ (middle) and SNR (left). . . . .	58
3.6	<b>Demonstration of real phantoms used for thickness estimation experimentation.</b> <b>Top:</b> Side view illustrations of the phantoms: (a) Round Ultem phantom, and (b) Aluminum phantom. <b>Bottom:</b> Time-domain signals from different layers of the (a) Ultem and (b) Aluminum phantoms. . . . .	59
3.7	<b>Thickness estimation results.</b> Mean Absolute Error (MAE, in mm) of the tested methods on the (a) Ultem and (b) Aluminum phantoms. In each figure, the left histogram shows average error on the thinnest layer, the right histogram shows the average error of the remaining layers. . . . .	60
3.8	<b>Thickness estimation visualization for layer physical phantoms.</b> Each pixel within the Time-of-Flight-Difference (TOFD) predictions (Gray scale images) indicates the estimated thickness for the spatial location. The color images are the corresponding TOFD error between the prediction and the ground truth. <b>Top:</b> Ultem phantom. <b>Bottom:</b> Aluminum phantom. . . . .	61

3.9	<p><b>A-scan prediction visualization.</b> <b>Left:</b> A-scan with US-CNN echo detection from the 0.5mm ring of the round Ultem phantom. <b>Right:</b> A-scan from the 0.1mm ring of the round Ultem along with detections of US-CNN, SMP and ISTA. . . . .</p>	62
3.10	<p><b>Error as a function of training distribution width.</b> Mean absolute errors of the four tested methods as a function of training distribution width. <b>(a)-(b) Round phantom.</b> MAE of (a) thinnest layer and (b) average of the rest of the layers. <b>(c)-(d) Rectangular phantom.</b> MAE of (c) thinnest layer and (d) average of the rest of the layers. Very large errors (full height bars) are measured when a method fails to find a second echo relevant for width estimation. . . . .</p>	63
4.1	<p><b>Method: anomaly-based representations for cell profiling.</b> <b>(A) Top:</b> Well-aggregated profiles. Left-to-right: The CellProfiler software was used to extract single cell morphology features from Cell Painting images, the single cell features were aggregated to a well-level profile. <b>Bottom:</b> Each plate (gray) includes control (cyan) and treated (color) wells (cell icon). Half of the control wells were used to train the in-distribution autoencoder (B) and the other wells were used for evaluation (C). <b>(B)</b> The in-distribution autoencoder was trained to minimize the reconstruction error of control wells. <b>(C)</b> The in-distribution autoencoder was used to reconstruct control and treated wells. The reconstruction errors were calculated for each well. <b>(D)</b> The reconstruction error of a treated well was standardized according to the reconstruction errors of the control wells that were not used for training. These standardized reconstruction errors formed the “anomaly”-based representation. Treatments that lead to high reconstruction errors (green), in respect to the controls’ reconstruction errors (blue), were defined as “hits” (threshold in dashed red). <b>(E)</b> Anomaly-based representation can be used for a variety of downstream applications. . . . .</p>	67
4.2	<p><b>Anomaly-based representations are more reproducible.</b> <b>(A)</b> Cartoon depicting the reproducibility determination. A treatment is defined as reproducible if the median pairwise correlation of its replicates (Replicate Correlation) is higher than the 90<sup>th</sup> percentile of the pairwise correlation. <b>(B)</b> Cartoon depicting the Percent Replicating score. The distribution of Replicate Correlations (green) versus the distribution of Random Pairs Correlations (black). The dashed red vertical line defines the reproducibility threshold of 90% of the random pairs distribution - a well to the right of this line (green cell icon) is defined as reproducible, and the fraction of reproducible treatments determines the Percent Replicating score. <b>(C)</b> Percent Replicating scores across datasets for the anomaly-based (left) and the CellProfiler representations (right). Distribution of Replicate Correlations (green - anomaly-based, red - CellProfiler-based). Distribution of Random Pairs Correlations (gray), zero correlation (dashed gray vertical line), reproducibility threshold (dashed red vertical line). <b>(D)</b> Venn diagram showing the number of reproducible treatments exclusive to the anomaly-based (green) or CellProfiler-based (orange) representations, and common to both representations (yellow). . . . .</p>	69



4.3	<p><b>Anomaly-based representations improve MoA classification.</b> (A) MoA classification workflow. Left-to-right: inclusion of compounds that met our reproducibility criteria (for either the CellProfiler or the Anomaly-based representations), followed by exclusion of MoAs with &lt; 5 compounds attributed to them. Training machine learning models using the anomaly-based versus the CellProfiler-based representations with cross-validation. (B) MoA classification results for anomaly-based versus CellProfiler-based representations, using the reproducible inclusion criteria with <math>(A \cup C \cup L)</math> or without <math>(C \cup L)</math> the anomaly-based representations. (C) F1-scores of anomaly-based (green) versus the CellProfiler-based (orange) representations trained on MLP and LR models for CDRP-bio (cpg0012-wawer-bioactivecompoundprofiling) and LINCS (cpg0004-lincs) datasets. Each dot represents an experiment and bars indicate confidence intervals. Red dashed line indicates the F1 random score. * - statistically significant (p-value &lt; 0.05) by Welsh t-test. P-values for CDRP-bio were 0.042 (LR) and 0.003 (MLP) and for LINCS 1.5e-10 (LR) and 0.003 (MLP) (D) MoA-specific F1 scores using the (better performing) LR model. Bold indicates MoAs that would not be included without reproducible compounds according to the anomaly-based representations. F1-scores lower than the random score for both representations (1/19 MoAs in CDRP-bio, and 9/55 MoAs in LINCS) were excluded from the figure to improve clarity. . . . .</p>	72
4.4	<p><b>MoA interpretation.</b> (A) Illustration of the explanation process of the anomalous features (orange on the right). This anomaly is explained by the combined alteration of multiple input features (yellow on the left). A group of well-level profiles of interest are passed through the network, and the features exhibiting the highest reconstruction errors are identified. For each investigated feature (orange square in output), the weights leading to its activation (green lines) are activated and the weights going from its CellProfiler representation in the input are deactivated (red lines). The well-level profiles are reintroduced into the network and the features that led to high reconstruction errors are found by using the autoencoder-based anomaly SHAP. These explanations can be pooled according to a treatment or MoA to provide the corresponding explanation. (B) Distributions of ATPase inhibitor top five features' z-scores for the CellProfiler (orange) and anomaly-based representations (green) ranked according to the CellProfiler median z-scores (top) and by anomaly median z-scores (bottom) in the CDRP-bio dataset. Features in bold are analyzed using the autoencoder-based anomaly SHAP in the following panels. (C-E) Explanations for selected (see text for justification) altered features in the ATPase inhibitor MoA in the CDRP-bio dataset. Each dot represents a replicate well, and the x-axis represents the SHAP values contributing to the errors. Negative values indicate an inverse relationship between the inspected feature and the input feature compared to the relationship in the control population. . . . .</p>	74
4.1	<p>Percent Replicating score as a function of the number of control wells used to train the in-distribution autoencoder. Dashed orange line - the percent replicating score of the CellProfiler representations. Green data points and shade show the percent replicating score mean and the standard deviation of the anomaly-based representations over 10 independent experiments. . . . .</p>	80
4.2	<p>Analysis of the complementary information shared between the anomaly-based, the CellProfiler, and the gene expression representations (L1000). <b>Top:</b> Venn diagram showing the number of reproducible treatments exclusive to the anomaly-based (green), CellProfiler-based (orange), and L1000-based (blue) representations. Treatments found reproducible by multiple representations are shown by the intersection of the different circles. <b>Below:</b> Percent Replicating scores for the L1000-based representations. Distribution of Replicate Correlations (blue), Random Pairs Correlations (gray), reproducibility threshold (dashed red vertical line). . . . .</p>	81

4.3	Genetic expression representations (L1000) complement the CellProfiler (orange) and anomaly-based (green) representations for MoA classification. Logistic Regression (LR) and Multi Layer Perceptron (MLP) MoA classification F1 performance on CellProfiler versus anomaly-based representations without (-L1000) or with (+L1000) the concatenation of the L1000 representation. Each dot represents a unique experiment and bars indicate 95% confidence intervals. Red dashed line indicates the F1 random score.	81
4.4	Granularity analysis in the AGP channel for the MoA of ATPase inhibitor in the CDRP-bio dataset. Feature z-scores distributions for CellProfiler (orange) and anomaly-based (green) representations. . . . .	82
4.5	Illustration of potential caveats of using weakly supervised (e.g., using the treatment as a weak label) (A) versus anomaly-based (B) representations. <b>(A)</b> A treatment representation (yellow) is implicitly becoming more similar to the control because it was guided away from another treatment (orange). <b>(B)</b> Anomaly-based representations are guided away from the control. . . . .	82
4.6	Scatter plot indicating the F1 score as a function of the number of treatments for each MoA in the CDRP-bio and LINCS dataset. The F1 scores were obtained with the better-performing LR model trained using the anomaly-based representations. . . . .	83

# List of Tables

3.1	Echo pattern parameter distributions in the simulated datasets. . . . .	57
4.1	<b>Datasets used in this study.</b> CDRP-bio (cpg0012-wawer-bioactivecompoundprofiling) [80], LINC637 (cpg0004-lincs) [83], LUAD (cpg0031-caicedo-cmvip) [91], TAORF (cpg0017-rohban-pathways) [113]. Treatment: Chemical compounds, or ORF overexpression. Controls: number of control wells. Treatments: number of distinct treatments. $N_r$ : median number of replicates per treatment. . . . .	68
4.2	<b>Reproducibility results.</b> Percent Replicating score for the CellProfiler (left column) versus the anomaly-based (middle column) representations. The union of treatments found reproducible by either representation is shown in the right column ( $A \cup C$ ). . . . .	70

# Abstract

This dissertation investigates the application of deep learning to three scientific domains and imaging acquisition sources: explainability for computer vision applied to natural images (Chapter 1), depth estimation for ultrasound applications (Chapter 2), and drug screening using biological microscopy (Chapter 3). The first chapter involves the development of statistical inference graphs for the interpretation of convolutional neural networks for computer vision based on generative probabilistic modeling. The second chapter focuses on detecting overlapping ultrasonic echoes using deep neural networks, improving depth estimation for non-destructive testing applications. The third chapter introduces an anomaly detection method for image-based cell profiling. These diverse investigations share the development and application of deep learning and computational modeling with contributions to each of the domains under investigation.

*Keywords:* Deep Learning, Convolutional Neural Networks, Artificial Intelligence, Computer Vision, AI explainability, Ultrasound, Cell Profiling

# Dissertation in a Nutshell

Advancements in science often arise from the convergence of diverse disciplines, each offering unique perspectives and methodologies. This doctoral dissertation explores the interpretation and application of deep learning to advance the state-of-the-art in three distinct domains. Deep learning has revolutionized the field of artificial intelligence, enabling machines to learn complex patterns and relationships from vast amounts of data. In recent years, deep neural networks have achieved remarkable success in various domains, including computer vision, natural language processing, and speech recognition. This success can be attributed to the ability of deep learning models to automatically learn hierarchical representations of data, capturing intricate patterns and abstractions that are difficult to hand-craft using traditional methods. This Ph.D. dissertation presents a comprehensive exploration of deep neural networks (DNN) with a focus on three subjects: interpretability of the inference process of DNNs' hidden layers activity based on probabilistic models (Chapter 1), detection of overlapping ultrasonic echoes using fully convolutional DNNs (Chapter 2), and the development of an anomaly detection method for high throughput microscopy-based cell phenotypic screening using deep autoencoders (Chapter 3). Throughout this interdisciplinary research, novel approaches and methodologies are introduced to advance our understanding of DNNs and address real-world challenges in computer vision, signal processing and biological screening. This variety of application domains was not planned, but rather a constraint caused by the decision of Aharon Bar-Hillel, my original Ph.D. mentor, to leave Ben-Gurion University and move to the industry. I moved to a new lab, which required me to shift my research focus. However, despite the disparate nature of the project, all three research topics share a common thread relying on deep neural networks, to address complex problems in their respective domains. Due to the differences between the chapters' problem domains, each chapter is covered by its own background, related work, and discussion.

The first chapter introduces a method for interpreting the hidden layer activity of convolutional neural networks (CNN) called SIGN. Motivated by the need for better understanding of the DNN reasoning process, generative probabilistic modeling is employed using Hidden Markov Models (HMM) and Gaussian Mixture Models (GMM) to model and visualize the network's intermediate layers activity. This approach involves clustering activity patterns of interest from both fully connected layers and spatial columns of convolutional layers using GMMs. Transition probabilities between clusters of consecutive layers are estimated using HMMs for fully connected layers and a maximum likelihood model for convolutional layers. The learned interpretation provides an explanatory inference graph describing the hierarchy of activity clusters most relevant for a prediction of a class or for explaining the decisions the network makes about specific images. The method

unveils valuable insights into the behaviour and information flow within DNNs, along with offering new techniques for increasing inference process transparency. An early version of this work was presented at the top computer vision conference *ECCV* in 2020 and its full version was published in the flagship computer vision journal *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* in 2022.

The second chapter focuses on detecting overlapping ultrasonic echoes in non-destructive testing. In thin layers, the time-of-flight disparity between two ultrasonic echoes often falls within the pulse width of the echoes, making it challenging to distinguish them. Traditional methods for separating overlapping echoes often struggle with high echo pattern variance and severe successive overlaps. To address this issue, a novel approach is proposed based on deep learning that treats the echo separation problem as a detection task. The method involves estimating the time of flight of echoes from a signal input using a fully convolutional neural network. Both the architecture of the CNN and the generation of data rely on prior knowledge of the underlying physics of the ultrasonic signal. The CNN architecture comprises long filters in the first layer to emulate the deconvolution of the signal back to its fundamental delta origin, followed by multiple layers featuring shorter filters to project the deconvolved activations to a space of clean and sparse sequences with minimal active delta functions. The network is trained on a simulated dataset constructed based on a physical model of ultrasound echoes, encompassing significant variations in echo parameters, noise, and overlaps. The proposed approach yielded improvements in depth resolution compared to traditional algorithms, as evidenced by simulations and real-world experiments, showcasing enhanced accuracy and resilience to echo variability. This research was published in the journal *Ultrasonics* in 2022.

In the third chapter, anomaly detection techniques for image-based cell profiling are explored. High-content image-based profiling is a powerful tool for identifying phenotypic differences in cell populations. Classical measurements of profiles falsely assume independence between morphological features, not fully capturing the complexity in cell organization. An alternative rooted in anomaly detection is proposed, which takes into consideration the interrelationships among different phenotypes. This enables the discovery of effects that each treatment has on cells while accounting for the relationships among various structure-related cell features. The suggested method capitalizes on the experimental setup of cell profiling, which typically includes a plethora of control population data (representing negative samples of inspected cells) and a limited number of samples for each tested treatment. A reconstruction-based autoencoder deep neural network is trained to encode a compressed phenotypic profile based on the distribution of the control data, subsequently decoding it to reconstruct the input profile. The vector of reconstruction errors for each treatment serves as the anomaly representation, with features exhibiting high reconstruction errors indicating altered dependencies within the feature representation. The superiority of the anomaly-based representations over traditional features is demonstrated across multiple high-content image-based datasets encompassing various cell types and treatments, showing improved reproducibility and effectiveness in the downstream task of mechanism of action (MoA) identification. Finally, to enable better explainability a method for interpreting the “cause” of the anomaly is introduced. This work

is in the final stages of writing a manuscript that will be posted in the upcoming days as a preprint on *bioRxiv*, and then submitted to a respected journal in the field of computational biology.

These chapters highlight the adaptability and efficacy of deep learning in tackling a wide range of challenges spanning various domains. Although each chapter is founded on different principles, they all share a common reliance on prior knowledge of the problem domain. Each deep learning model propels advancements within its respective field by harnessing a prior understanding of the problem domain. The initial chapter is grounded in probabilistic principles related to the function of deep networks, which are pivotal for modeling the hidden layers effectively. The second chapter draws upon prior knowledge of the underlying physics of the ultrasonic echo, essential for constructing the simulation dataset and CNN architecture. The last chapter builds upon the underlying structure of biological experiments as the foundation for the anomaly detection framework. In times where deep models are often viewed as a "magic solution", this might serve as a reminder that employing these powerful models is insufficient for advancing the state-of-the-art; it is imperative to rely on deep-rooted foundations within the problem domain to drive meaningful progress.

# SIGN: Statistical Inference Graphs Based on Probabilistic Network Activity Interpretation

## Publications:

- Konforti, Y. <sup>‡</sup>, **Shpigler, A.** <sup>‡</sup>, Lerner, B., & Bar-Hillel, A. (2020). Inference graphs for CNN interpretation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXV 16* (pp. 69-84). Springer International Publishing.
- Konforti, Y. <sup>‡</sup>, **Shpigler, A.** <sup>‡</sup>, Lerner, B., & Bar-Hillel, A. (2022). SIGN: Statistical inference graphs based on probabilistic network activity interpretation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3), 3783-3797.

## Abstract:

Convolutional neural networks (CNNs) have achieved superior accuracy in many visual-related tasks. However, the inference process through a CNN's intermediate layers is opaque, making it difficult to interpret such networks or develop trust in their operation. In this paper, we introduce SIGN method for modeling the network's hidden layer activity using probabilistic models. The activity patterns in layers of interest are modeled as Gaussian mixture models, and transition probabilities between clusters in consecutive modeled layers are estimated to identify paths of inference. For fully connected networks, the entire layer activity is clustered, and the resulting model is a hidden Markov model. For convolutional layers, spatial columns of activity are clustered, and a maximum likelihood model is developed for mining an explanatory inference graph. The graph describes the hierarchy of activity clusters most relevant for network prediction. We show that such inference graphs are useful for understanding the general inference process of a class, as well as explaining the (correct or incorrect) decisions the network makes about specific images. In addition, SIGN provide interesting observations regarding hidden layer activity in general, including the concentration of memorization in a single middle layer in fully connected networks, and a highly local nature of column activities in the top CNN layers.

---

<sup>‡</sup> Equal contribution.

<sup>‡</sup> Equal contribution.



## 2.1 Introduction

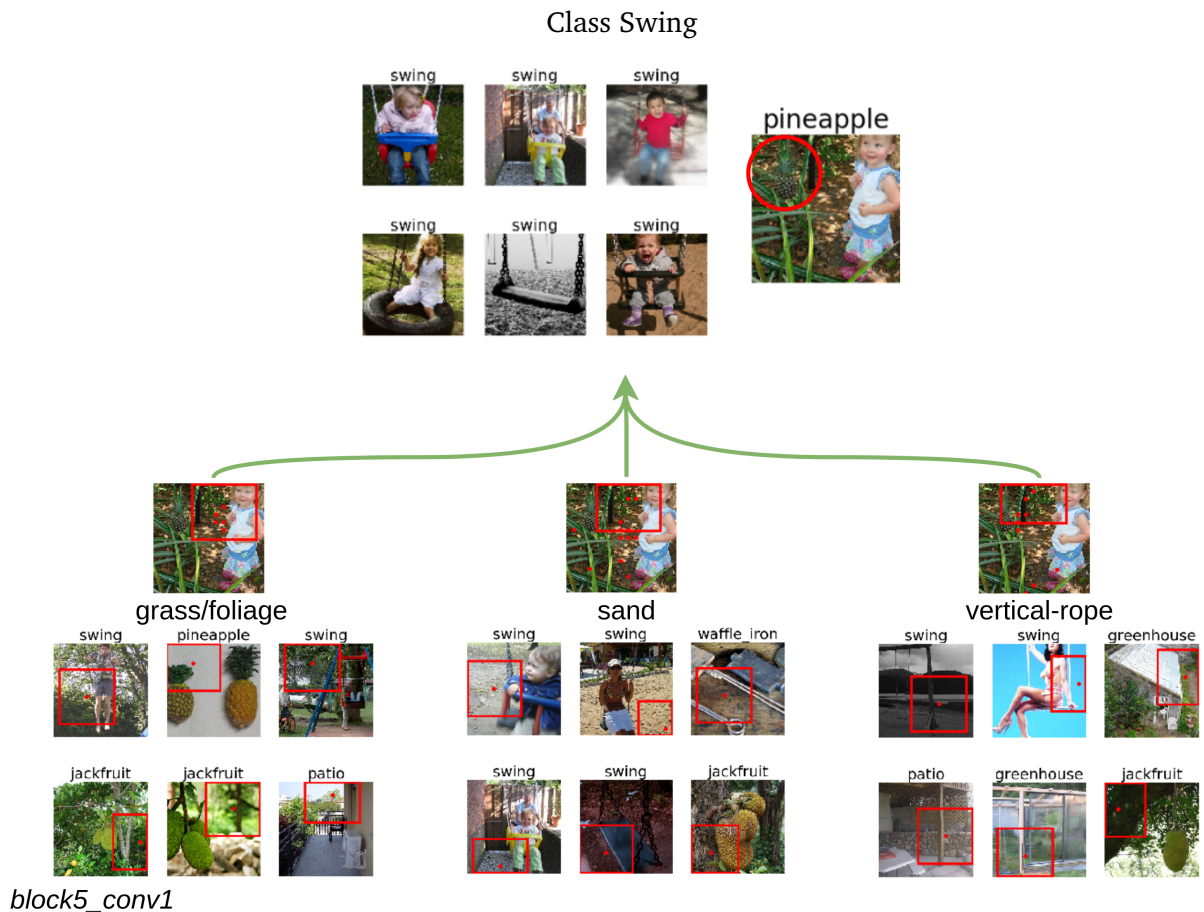
Thanks to their impressive performance, convolutional neural networks (CNNs) are the leading architecture for tasks in computer vision [1, 2, 3]. However, current deep-learning methods suffer from poor interpretability of the inference process conducted by their hidden layers. Due to their end-to-end training and complex architecture, the reasoning behind their decision-making process is hard to interpret. The lack of transparency undermines the trustworthiness and reliability of deep networks, especially in sectors where explainability is deemed crucial, such as the healthcare and autonomous-driving sectors. From this need emerged the research field of Explainable Artificial Intelligence (XAI), aimed at promoting intuitive, human-understandable explanations of AI decision-making process [4, 5, 6].

A majority of XAI techniques focus on explaining local decisions for specific images or neurons [7, 8, 9]. These methods are popular due to their simplicity and ease of use. However, this kind of interpretation can easily miss important factors in the decision chain leading to a specific prediction. Understanding CNN reasoning by decomposing it into layer-wise stages can provide insights about cases of failure, and reveal weak spots in the network architecture, training scheme, or data collection mechanism. In turn, these insights can lead to more robust networks, and allow us to develop more trust in CNN decisions. Some efforts are made in this direction in recent years [10, 11], however it remains a pre-mature field with many challenges and opportunities ahead.

In this work, we seek to enable a better human-understanding of the deep network inference process. As we see it, this mission requires facing several challenges:

**(1)** Transforming a distributed high dimensional representation into a discrete representation amendable to human reasoning. Deep networks operate through a series of distributed layer representations, manifested by a single activation vector in fully connected (FC) layers and a collection of spatial activation vectors (i.e., columns) in convolutional layers. Human language, however, is made up of discrete symbols, i.e., words, having meaning grounded by their reference in the world of objects and their interrelations. Given the richness of a distributed representation and the sheer size of modern networks, discretizing an internal representation may require thousands of visual words. Hence, **(2)** Selecting a relevant subset of visual words and connections for a specific analysis task (e.g., class-specific or image-specific analysis) is inherent to this endeavor. **(3)** Development of a visualization system that enables understanding of the visual words and their interrelations, while taking into account the human perceptual and cognitive limitations and display capacity [12].

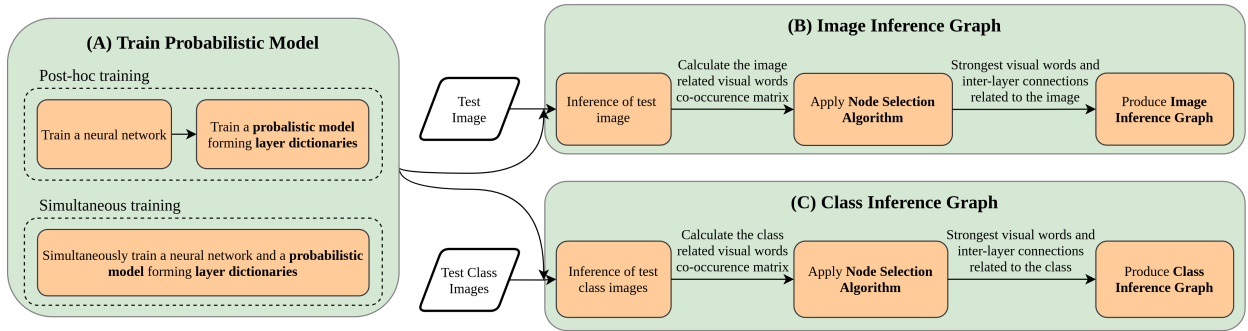
In this paper we introduce SIGN, a statistical explanation framework of the inference process conducted by a deep neural network, based on probabilistic interpretation of network activity. A demonstration of the SIGN framework output is shown in Fig. 2.1. In this example, the framework provides an interpretation of a false prediction made by VGG-16, of a pineapple image to class



**Figure. 2.1: Explaining network errors with SIGN.** An ILSVRC-2012 [13] dataset image of a pineapple (red-circled), falsely classified by VGG-16 to the “swing” class. This is a partial inference graph with “swing” as the inferred class at the top, and the 3 most influential visual words from a high level layer (block5\_conv1 layer), forming it. The 3 visual words show *what* the network found to be the most important features for classification of the inspected image. Each such word is represented using 6 images containing it, with the visual word itself in a red rectangle. The analyzed image is presented above each visual word, with red dots showing *where* this visual word is found. As seen in the graph, the image is classified based on 3 “swing”-related visual words describing “rope”, “sand” and “grass”. The full inference graph is seen in Fig. 2.9 Section 2.4.3.

“swing”. SIGN points out *what* are the crucial visual words for the network’s prediction and *where* these visual words occur in the image.

The SIGN framework, illustrated in Fig. 2.2, is composed of a few stages. A *probabilistic model* is learned on top of the network architecture, and can be trained simultaneously with network training or post-hoc, to generatively describe network layer activity behavior and layer-wise dependencies. Activity vectors in each layer are modeled as arising from a multivariate Gaussian mixture model (GMM). Layer activity in FC layers, or spatial location activity in convolutional layers, is associated with one of  $K$  clusters (GMM components), each representing a visual word. Together, all visual words within the same layer forming the layer’s dictionary. Connections between visual words of consecutive layers are modeled using conditional probabilities. For a multi layer perceptron (MLP) network, a full model with efficient inference can be obtained using a hidden Markov model (HMM). For convolutional layers, each spatial location has its own hidden variable. A full exact



**Figure. 2.2: Illustration of SIGN framework flow.** (A) Modeling neural network layers as arising from a probabilistic generative model, and forming a visual word dictionary to each modeled layer. Training the generative model can be simultaneously with the network or post-hoc. (B) Forward pass through the modeled network on a subset inputted test images. Calculate the co-occurrence matrix statistics for all visual words that appear in the image. Apply Node Selection Algorithm on the matrix to obtain the most explanatory visual words. Produce an image inference graph of words as nodes and their connection strengths as edges. (C) The same as (B) but with a subset of analyzed images that represent a class predicted by the network.

inference is thus infeasible due to the high induced width of the resulting graphical model. Instead, in the suggested model dependencies between neighboring words are ignored, and dependencies among visual words in consecutive layers are described using conditional probability tables. Given a selected subset of images to be explained, the decision process of the network can be described using an inference graph. The inference graph levels representing the visual words used to explain the subset of images in different layers of the network and their weighted connections. As the full graph may contain thousands of visual words in all network layers, a useful explanation has to find informative subgraph containing the most explanatory words for clarifying the network decision. Therefore, we suggest a maximum likelihood based *Node Selection Algorithm* for finding such informative subgraphs. Finally, based on the node selection algorithm we visualize the network decision process explanation as an *Inference Graph*.

With SIGN we aim to provide an inference investigation tool for deep models with the following contributions:

- **A class-specific inference graph:** The class inference graph provides a succinct summary of the inference process toward a specific class, as it progresses through the network layers. The connections between visual words discovered by SIGN in consecutive layers provide clear insights into the feature aggregation process for this class.
- **An image-specific inference graph:** The inference graph for a specific image highlights the visual words most contributing to a class decision on this image (see Fig. 2.1). Such a graph is highly useful as a debugging tool to analyze network failures as it enables finding *what* are the main features leading to a false class prediction, which are the layers these features appear, and *where* do they appear in the input image.

- **Differences in layer activity behavior between MLP networks and CNNs:** Our model demonstrates significant differences in activity behavior between the two network types as inference progresses through the network. For MLP networks, visual words gradually converge, from multiple input-related words to unique class-related words. In contrast, CNN behavior remains local and diverse even at the uppermost layers, with each class represented by a combination of distinct multiple words.
- **Overfitting capacity of a single layer:** Following Zhang et al. [14], we analyze a case of extreme overfit by learning with random labels. Our model discovers that for MLP network forced to such extreme overfit, the overfitting transformation is concentrated in a single (the middle) hidden layer. In such a case, our suggested tools enable, for the first time as far as we are aware, characterizing *where* in the network overfit occurs.

## 2.2 Related Work

An important contribution of XAI methods involves explaining neural networks decisions and internal mechanisms. Methods can be generally categorized by the scope of their explanations: instance-level explanation methods, and internal network behavior explanation. We review below some of the important works in this vast domain. For a more comprehensive review of XAI methods, readers are referred to recent surveys [4, 5, 6].

### 2.2.1 Instance-Level Explanations

A favorable group of methods for network interpretation involves local explanations of specific data instances or a network component. This is mainly achieved via back propagation techniques [15, 16, 7, 9, 8] and perturbation methods [17, 18, 19]. Back propagation based methods use the gradient information back-propagated from the output prediction layer back to the input layer. Among the popular techniques are activation maximization and attribution. With activation maximization [15, 16, 7], the input space is randomly initialized and optimized to maximize the score of a specific feature, producing an image of what this feature is looking for. Zeiler et al. [18] introduced deconvolution layers showing which input pattern originally induced a given activation. This is done by creating an input map that keep the examined activation values, and set all other to zero. Attribution methods [8, 9, 20] highlight the input regions that are most valuable for the network prediction for a specific image. This technique was first introduced as saliency maps by Simonyan et al. [8], that shows gradients of a class prediction with respect to an input image. Zhou et al. [20] suggested Class Activation Mapping (CAM), a simple modification to global average pooling that reveals how regions in an input image are correlated to a specific class with a single forward-pass. This idea was later enhanced by Grad-CAM [9] that generalized CAM method to a broader types of CNN models.

Perturbation methods search for the correlation between the input and the output while the input image is changing (i.e., perturbing). This is mostly done by occluding image pixels group and observing the networks prediction changes. LIME [17] uses superpixel method to occlude pixels group, then they approximating a linear model for the network behaviour which can be interpreted. In [19], the authors search for the perturbation mask that gain the maximal effect on the network's output among all masks.

While such visualizations producing good local explanation and mostly are easy to implement, they are anecdotal and insufficient for understanding the full network's reasoning. In addition, these methods are sensitive to input noise [21] and hyper-parameter tuning [22]. As recently shown in [23], they are not more effective for user understanding than showing the nearest training set examples in most cases.

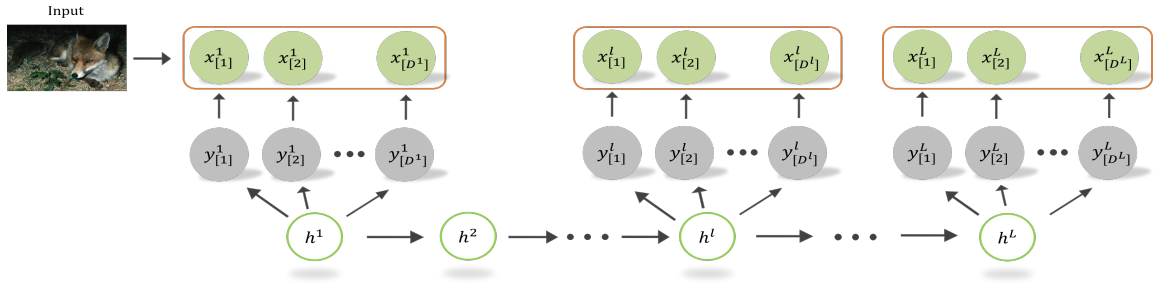
Instance-level explanations can be enhanced with the addition of network architecture context, to provide an hierarchical analysis on how the decision is propagated through the network's layers. Olah et al. [24] proposed a tool for visualizing the network path for a single image. They decomposed each layer's activations into neuron groups using matrix factorization, and visualized each group, both by activation maximization and attribution. Then, groups from consecutive layers were connected to form a graph structure by their weight connection strength.

The SIGN method offers instance level explanations with context of network architecture, showing the critical features in each layer leading to classification. This forms a decision-chain of the instance along the network layers. Feature visualizations are created based on aggregated information from the entire training set. Visual words are represented by image patches, thus keeping the context of real-world objects.

## 2.2.2 Internal Network Behavior Explanation

Some explanation efforts are aimed at promoting transparency of hidden layers behavior. One approach tries to quantify the features by their roles across different layers and provide a layer-wise summary of the model. Bau et al. [25] defined six types of semantic patterns (colors, textures, materials, parts, objects, and scenes) identified by CNNs, and labeled image pixels accordingly. In their later work [11], they increased the number of semantic concepts roles, thus gained more fine grain feature groups. In [26], the authors present a model explanation summary where they grouped together images with similar explanations and sub-grouped them based on similar features that explains them. They evaluate explanation based on information theory.

Another form of network-centric XAI techniques aim to look beyond individual features and provide a layer-wise description of inner-decision mechanism operating to produce the desired outputs. Some methods do so by enforcing hidden features to represent meaningful representations [27, 28]. Similar to [25], Zhang et al. [27] aimed to associate filters with object part but in this work



**Figure. 2.3:** A graphical model for MLP networks. Each orange rectangle is a layer activation vector after the ReLU operation. Activation  $x^l[d]$  of neuron  $d$  in layer  $l$  is assumed to be generated from a rectified Gaussian density, resetting values lower than zero to zero.  $y^l[d]$  is the parent of  $x^l[d]$ , describing the original Gaussian density before it was rectified.  $h^l$  is a hidden variable generating the hidden vector of multivariate Gaussians  $Y^l$ .  $h^l$  takes  $K^l$  different states, creating a mixture of multivariate Gaussians for layer  $l$ .

they enforce the network filters to do so. Chen et al. [28] impose interpretability by encouraging spatial columns of the topmost conv layer to represent part prototypes of a specific class, with each prototype equated with a spatial region patch of the input image belong to the same class. Loss terms are added to enforce between-class separation, and inner-class property.

Some works wish to explain the overall internal model behaviour without alter the network representations. CNNVis [29] provide a model summary, with neurons in each layer clustered to form groups having similar activity patterns. A graph between neuron clusters of subsequent layers is then formed based on the average weight strengths over the cluster's neurons. Hohman et al. [10] proposed Summit, an interactive visualization tool presenting class attribution graph. This graph visualize the aggregated top channel activations in each layer across all images within the same class. Connections between channels in consecutive layers are quantified based on the influence of the former channel on the latter. This graph reveals interesting and unexpected connections between channels associated with different classes across layers.

The above methods offer a wider perspective on how the intelligent systems work. Inspired by these works, we suggest a novel approach for hidden layers representation. We go beyond specific features and simple clustering techniques to offer an holistic, generative model for full network explanation.

## 2.3 Method

Inference graphs for an MLP, for which a full graphical model can be suggested, are presented in Section 2.3.1. The more general case of a CNN is discussed in Section 2.3.2, and its related graph-mining algorithm in Section 2.3.3. Models can be trained on the full set of network layers or on a subset, indexed by  $l \in \{1, \dots, L\}$ .



### 2.3.1 Inference Graphs for MLPs

A network composed of FC layers can be modeled by a single probabilistic graphical model based on the following assumptions: (a) The activity of a layer can be modeled by a single mixture model. (b) Conditional independence holds between the activity of layer  $l$  and activities of layers preceding  $l - 1$  given the activity of layer  $l - 1$ . (c) Layer activity is generated by a rectified normal distribution [30], censored at zero according to ReLU operation. For such a network, the activity of hidden layers is modeled by an HMM structure, enabling closed-form inference. The model structure is shown in Fig. 2.3.

For the  $l$ th FC layer with  $D^l$  neurons, denote the activation vector as  $x^l = (x^l[1], \dots, x^l[D^l]) \in \mathbb{R}^{D^l}$ . The distribution of  $x^l$  is modeled using a mixture of  $K^l$  hidden states (i.e., clusters) with a discrete hidden variable  $h^l \in \{1, \dots, K^l\}$  denoting the cluster index. To model the ReLU operation, each neuron activation  $x^l[d]$  is generated from a rectified Gaussian distribution. The conditional probability  $P(x^l|h^l)$  is hence assumed to be a rectified multivariate Gaussian distribution with a diagonal covariance matrix. Connections between hidden variables in consecutive layers, are modeled by a conditional probability table (CPT)  $P(h^l|h^{l-1})$ .

Using this generative model, an activity pattern for the network is sampled by three steps. First, a path  $(h^1, \dots, h^L)$  of hidden states is generated according to the transition probabilities

$$P(h^l = k|h^{l-1} = k') = t_{k,k'}^l \quad (2.1)$$

where  $t^l \in \mathbb{R}^{K^l \times K^{l-1}}$  is a learned CPT. For notation simplicity, we define  $h^0 = \{\}$ , so  $P(h^1|h^0)$  is actually  $P(h^1)$  parametrized by  $P(h^1 = k) = t_k^1$ . After path generation, "pre-ReLU" Gaussian vectors  $(y^1, \dots, y^L)$ , with  $y^l \in \mathbb{R}^{D^l}$ , are generated based on the chosen hidden variables. A single variable  $y^l[d]$  is formed according to

$$P(y^l[d]|h^l = k) \sim \mathcal{N}(y^l[d]|\mu_{d,k}^l, \sigma_{d,k}^l), \quad (2.2)$$

where  $\mu_{d,k}^l$  and  $\sigma_{d,k}^l$  are the mean and standard deviation of the  $d$ th element in the  $k$ th component of layer  $l$ . Since the observed activity  $x^l[d]$ , generated as  $x^l[d] = \max(y^l[d], 0)$ , is a deterministic function of  $y^l[d]$ , its conditional probability  $P(x^l[d]|y^l[d])$  can be written as

$$P(x^l[d]|y^l[d]) = \begin{cases} \delta_{x^l[d]=y^l[d]} & , y^l[d] > 0 \\ \delta_{x^l[d]=0} & , y^l[d] \leq 0 \end{cases} \quad (2.3)$$

with  $\delta_{(x=c)}$  as the Dirac delta function concentrating the distribution mass at  $c$ .

The full likelihood of the model is given by

$$P(X, Y, H|\Theta) = \prod_{l=1}^L P(h^l|h^{l-1})P(y^l|h^l)P(x^l|y^l), \quad (2.4)$$

where  $P(h^l|h^{l-1})$ ,  $P(y^l|h^l)$ , and  $P(x^l|y^l)$  are stated in (2.1), (2.2), and (2.3), respectively.  $Y, H, X$  are tuples representing their respective variables across all layers, e.g.,  $H = \{h_l\}_{l=1}^L$ . The set of parameters  $\Theta$  learned to optimize the model likelihood is

$$\Theta = \left\{ \{t_{k,k'}^l\}_{l=1, k=1, k'=1}^{L, K^l, K^{l-1}}, \{\mu_{d,k}^l\}_{l=1, d=1, k=1}^{L, D^l, K^l}, \{\sigma_{d,k}^l\}_{l=1, d=1, k=1}^{L, D^l, K^l} \right\}. \quad (2.5)$$

**Training algorithm:** In [31], the EM formulation was suggested for training a mixture of censored Gaussians. We extended this idea to the HMM formulation in an online setting. Following [32], the online EM algorithm tracks the sufficient statistics using running averages, and updates the model parameters using these statistics. We train with mini-batches, which enables scalable learning for large-scale networks. Training iterates between updating the running averages of the sufficient statistics (an online approximation of the E-step), and updating the parameters based on these averages (the M-step). This procedure is shown [32] to be consistent (i.e., finding a stationary point of the data log likelihood with probability 1) and asymptotically efficient.

**Update equations:** In a batch EM formulation, model parameters are updated based on sample statistics of interest. Each statistic is defined as the sample average  $\frac{1}{N} \sum_{i=1}^N f(X_i)$  for a function  $f(x)$  of interest. In the online setting, for each such function  $f(x)$ , an online sample estimator is kept, denoted here by  $\langle f(X) \rangle$ . Given a batch of examples  $\{X_i\}_{i=1}^B$  and adaptation parameter  $\alpha > 0$ ,  $\langle f(X) \rangle$  is updated in iteration  $q + 1$  by

$$\langle f(X) \rangle_{q+1} = (1 - \alpha) \langle f(X) \rangle_q + \frac{\alpha}{B} \sum_{i=1}^B f(X_i) \quad (2.6)$$

The tracked sufficient statistics are used in the update of the model parameters as follows:

- The transition probability between hidden states  $t_{k,k'}^l$  update is given by

$$t_{k,k'}^l = \frac{\langle P(h^l = k, h^{l-1} = k'|X, \Theta) \rangle}{\sum_{k=1}^{K^l} \langle P(h^l = k, h^{l-1} = k'|X, \Theta) \rangle}. \quad (2.7)$$

The average joint distribution of clusters from consecutive layers  $\langle P(h^l = k, h^{l-1} = k'|X, \Theta) \rangle$  is a tracked statistic, computed for each example using the forward-backward algorithm[33]. For the first layer,  $t_k^1$  is updated analogously using  $t_k^1 = \langle P(h^1 = k|X, \Theta) \rangle$ .



- The mean  $\mu_{d,k}^l$  for an estimated Gaussian before rectification (dropping the  $l$  index for notation convenience) is

$$\mu_{d,k} = \frac{\langle P(h = k|x[d], \Theta) \cdot \hat{y}[d] \rangle}{\langle P(h = k|x[d], \Theta) \rangle} \quad (2.8)$$

with  $\hat{y}[d]$  defined by

$$\hat{y}[d] = \begin{cases} x[d], & x[d] > 0 \\ M_1(\mu_{d,k}, \sigma_{d,k}), & x[d] = 0 \end{cases}$$

The new mean is a weighted average of all examples'  $y$  activities, with each example contributing based on its probability to belong to the cluster. When  $x[d] = 0$ , the value of the activity prior to the ReLU operation is estimated using the first moment of a rectified Gaussian  $M_1(\mu_{d,k}, \sigma_{d,k})$ , with  $\mu_{d,k}$  and  $\sigma_{d,k}$  values are taken from the previous iteration.  $M_1(\mu_{d,k}, \sigma_{d,k})$  has a closed form solution [31] for known mean and variance:

$$M_1(\mu, \sigma) = \int_{-\infty}^0 x \cdot G(x|\mu, \sigma) dx = \mu - \sigma \frac{(G(-\frac{\mu}{\sigma}|0, 1))}{(C(-\frac{\mu}{\sigma}|0, 1))} \quad (2.9)$$

where  $G(-\frac{\mu}{\sigma}|0, 1)$  and  $C(-\frac{\mu}{\sigma}|0, 1)$  are the density and cumulative values of the normal distribution at  $-\frac{\mu}{\sigma}$ . Since  $\hat{y}[d]$  has two cases, two running statistics are tracked for the computation of the nominator in Eq. 2.8:  $\langle P(h = k|x[d], \Theta) \cdot x[d]1_{x[d]>0} \rangle$  and  $\langle P(h = k|x[d], \Theta) \cdot 1_{x[d]=0} \rangle$ .

- The std of an estimated Gaussian density before rectification  $\sigma_{d,k}^l$ , dropping the  $l$  index, is updated by

$$\sigma_{d,k}^2 = \frac{\langle P(h = k|X, \Theta)(\hat{y}[d] - \mu_{d,k})^2 \rangle + R_{d,k}}{\langle h = k|X, \Theta \rangle}. \quad (2.10)$$

This formula can be seen as a weighted sum-of-squares and a correction factor

$$R_{d,k} = \langle P(h = k|x[d], \Theta) \cdot 1_{x[d]=0} \rangle \cdot (M_2(\mu_{d,k}, \sigma_{d,k}) - M_1(\mu_{d,k}, \sigma_{d,k})^2). \quad (2.11)$$

The term  $M_2(\mu_{d,k}, \sigma_{d,k})$  is the second moment of a censored Gaussian distribution, which also has a closed form solution [31]:

$$M_2(\mu, \sigma) = \int_{-\infty}^0 x^2 G(x|\mu, \sigma) dx = \mu^2 + \sigma^2 - \sigma\mu \frac{(G(-\frac{\mu}{\sigma}|0, 1))}{(C(-\frac{\mu}{\sigma}|0, 1))}. \quad (2.12)$$

### 2.3.2 Inference Graphs for CNN

In a CNN, the activation output of the  $l$ -th convolutional layer is a tensor  $X^l \in \mathbb{R}^{H^l \times W^l \times D^l}$ , where  $H^l$ ,  $W^l$ , and  $D^l$  correspond to the height, width, and number of maps, respectively. We consider the activation tensor as consisting of  $H^l \times W^l$  spatial column examples,  $x_p^l \in \mathbb{R}^{D^l}$ , located at

$p = (i, j) \in \{\{1, \dots, H^l\} \times \{1, \dots, W^l\}\}$ , and wish to model each such location as containing a separate visual word from a dictionary shared by all locations. The number of hidden variables (one per location) is much larger than in an FC layer (where a single hidden variable per layer was used), and their connectivity pattern across layers is dense, leading to a graphical model with high induced width, but with infeasible exact inference [34]. Hence, we turn to simpler model and training techniques, that are scalable to the size and complexity of CNNs. In this model, activities of different layers are modeled independently, each using a Gaussian mixture model, and transition probabilities between clusters in consecutive layers are modeled a-posteriori (i.e., they are not part of the generative model).

**Layer dictionaries:** Similarly to the full activity vector in FC layers, the activity of a spatial column  $x_p^l$  is described as arising from a GMM of  $K^l$  clusters, regarded as visual words forming the layer dictionary. Using a training image set  $S_T = \{(I_n, y_n)\}_{n=1}^{N_T}$ , a GMM is trained independently for each layer of interest. While each location in layer  $l$  has a separate hidden random variable  $h_p^l$ , the GMM parameters are shared across all the spatial locations of that layer, i.e., a single GMM is trained using all spatial columns of layer  $l$  and all training images. Training is done using a gradient descent procedure on mini-batches, enabling straight-forward GPU implementation. After model training, the activity tensor of layer  $l$  for a new example  $I$  can be mapped into a tensor  $P \in \mathbb{R}^{H^l \times W^l \times K^l}$  holding  $P(h_p^l(I) = k)$ . We say that the visual word  $h_p^l(I) = k^*$  (an activation column of image  $I$  in position  $p$  is assigned to cluster  $k^*$ ) iff  $k^* = \operatorname{argmax}_k P(h_p^l(I) = k)$ . Accordingly, visual word  $k$  in layer  $l$  is the cluster  $C_k^l = \{(I, p), I \in S_T : h_p^l(I) = k\}$  containing activations over all positions for all images in the training set  $S_T$ , where cluster  $k$  has the highest  $P(h_p^l(I) = k)$ .

When the CNN also contains FC layers, these can be modeled using a GMM trained on the layer's activity vectors. This can be regarded as a degenerate case of convolutional layer modeling, where the number of spatial locations is one. Specifically, the output layer ( $X^L$ ) of the network, containing  $B$  class pseudo probabilities (output neurons after softmax), is modeled using a GMM of  $B$  components (the same amount as classes was set for this layer GMM components). This GMM is not trained, and instead is fixed such that  $\mu_{d,b} = 1$  for  $d = b$  and 0 otherwise, and a constant variance parameter of  $\sigma_{d,b} = 0.1$ . In this setting, cluster  $b$  of the output layer contains images that the network predicts to be of class  $b$ .

**Probabilistic connections between layer dictionaries:** Transition probabilities between visual words in consecutive layers are modeled a-posteriori. For two consecutive modeled layers  $l'$  and  $l$  ( $l' < l$ ), the receptive field  $R(p)$  of location  $p$  in layer  $l$  is defined as the set of locations  $\{q = p + o : o \in O\}$  in layer  $l'$  used in the computation of  $x_p^l$ .  $O$  is a set of  $\{(\Delta x, \Delta y)\}$  integer offsets. Using a validation sample  $S_V = \{I_n\}_{n=1}^{N_V}$ , we compute for each two consecutive modeled layers  $l$  and  $l'$  the co-occurrence matrix  $N \in \mathbb{M}^{K^l \times K^{l'}}$  between the visual words these two dictionaries contain,

$$N(k, k') = \left| \{(I_n, p, q) : h_p^l(I_n) = k, h_q^{l'}(I_n) = k', q \in R(p)\} \right|. \quad (2.13)$$

Using  $N$ , we can obtain the following first and second order statistics:

$$\hat{P}(h^l = k) = \frac{\sum_j N(k, j)}{\sum_{i,j} N(i, j)} \quad (2.14)$$

$$\begin{aligned} \hat{P}(h'_q = k' | h_p^l = k, q \in R(p)) &= \frac{N(k, k')}{\sum_j N(k, j)} \\ &= \frac{|\{(I_n, p, q) : h_p^l(I_n) = k, h'_q(I_n) = k', q \in R(p)\}|}{|O| \cdot |\{(I_n, p) : h_p^l(I_n) = k\}|} \\ &= \frac{1}{|O|} \sum_{o \in O} \hat{P}(h'_{p+o} = k' | h_p^l = k) \end{aligned} \quad (2.15)$$

The transition probabilities as defined above are abbreviated in the following discussion to  $\hat{P}(h' = k' | h^l = k)$ . As defined, these probabilities are averaged over specific positions in the receptive field, since modeling of position-specific transition probabilities separately would lead to proliferation in the parameter number.

**Training algorithm:** The GMM parameters  $\Theta^l$  of layer  $l$  are trained by associating a GMM layer to each modeled layer of the network. Since we do not wish to alter the network's behavior, the GMM gradients do not propagate towards lower layers of the network. We considered two different optimization approaches for training  $\Theta^l$ :

- *Generative loss*—The optimization objective is to minimize the negative log-likelihood function:

$$\mathcal{L}_G(X^l(I_n), \Theta^l) = - \sum_{p \in A} \log \sum_{k=1}^{K^l} \pi_k^l G(x_p^l(I_n) | \mu_k^l, \Sigma_k^l) \quad (2.16)$$

with

$$\Sigma_k^l = \begin{bmatrix} \sigma_{1,k}^l & \dots & \sigma_{D^l,k}^l \end{bmatrix} \times I_{eye}$$

where  $G$  is the Gaussian distribution function,  $\pi_k^l$  is the mixture probability of the  $k$ 'th component in layer  $l$ , and  $I_{eye} \in \mathbb{M}^{D^l \times D^l}$  is the identity matrix.

- *Discriminative loss*—The probability tensor  $P$  is summarized into a histogram of visual words  $Hist^l(X^l(I_n)) \in \mathbb{R}^{K^l}$  using a global pooling operation. A linear classifier  $\mathcal{W} \cdot Hist^l(I_n)$  is formed and optimized by minimizing a cross entropy loss, where  $\mathcal{W}$  is the classifier weights vector

$$\mathcal{L}_D(X^l(I_n), \Theta^l, y_n) = - \log P(\hat{y}_n = y_n | \mathcal{W} \cdot Hist^l(X^l(I_n), \Theta^l)). \quad (2.17)$$

Here,  $y_n$  is the true label of image  $I_n$  and  $\hat{y}_n$  is the predicted output after a softmax transformation.

For the sake of clarity, these two losses are separate approaches for optimizing the model's parameters. Empirical comparison between these two approaches is given in Section 2.4.3.

For ImageNet-scale networks, full modeling of the entire network at once may require thousands of visual words per layer. Training such large dictionaries is not feasible with current GPU memory limitations (12GB for a TitanX). Our solution is to train a class-specific model, explaining network behavior for a specific class  $b$  and its “neighboring” classes, i.e., all classes erroneously predicted by the network for images of class  $b$ . The set of neighboring classes is chosen based on the network’s confusion matrix computed on the validation set. The model is trained on all training images of class  $b$  and its neighbors.

### 2.3.3 Node Selection Algorithm

Consider a graph in which column activity clusters (i.e., visual words)  $\{C_k^l\}_{l=1, k=1}^{L, K^l}$  are the nodes, and transition probabilities between clusters of consecutive layers quantify edges between the nodes. Typically, this graph contains thousands of nodes and, thus, is not feasible for human interpretation. However, specific subgraphs may have high explanatory value. Specifically, nodes (clusters) of the final layer  $C_k^L$  in this graph represent images for which the network predicted a class  $k$ . To understand this decision, we evaluate clusters in the previous layer  $C_{k'}^{L-1}$  using a score based on the transition probabilities  $P(h^L = k | h^{L-1} = k')$ . The step of finding such a set of “explanatory” clusters in layer  $L - 1$  is repeated to lower layers. Below, we develop a suitable iterative algorithm. Given a validation subset of images  $\Omega = \{I_n\}_{n=1}^N$ , it outputs a subgraph of the nodes that most “explain” the network decisions on  $\Omega$ , where “explanation” is defined in the maximum-likelihood sense. We first explain node selection for a single visual word in a single image, and then extend this notion to a full algorithm operating on multiple visual words and images.

#### 2.3.3.1 Explaining a Single Visual Word

Consider an instance of a single visual word  $h_p^l(I) = s$ , derived from a column activity location  $p$  in layer  $l$  for image  $I$ . Given this visual word, we look for the visual words in  $R(p)$  most contributing to its likelihood, given by (omitting the image notation  $I$  in  $h_p^l(I)$  for brevity):

$$\begin{aligned}
 P\left(h_p^l = s \mid \{h_q^l : q \in R(p)\}\right) &= \\
 & \frac{P\left(\{h_q^l : q \in R(p)\} \mid h_p^l = s\right) \cdot P(h_p^l = s)}{P\left(\{h_q^l : q \in R(p)\}\right)} \\
 & \approx \frac{\prod_{q \in R(p)} P(h_q^l \mid h_p^l = s) \cdot P(h_p^l = s)}{\prod_{q \in R(p)} P(h_q^l)}.
 \end{aligned} \tag{2.18}$$

In the last step, two simplifying assumptions were made: conditional independence over locations in the receptive field (nominator) and independence of locations (denominator). Taking the logarithm, we decompose the expression and see the contribution of visual words to the likelihood:

$$\begin{aligned} & \underbrace{\log P(h_p^l = s)}_{\text{constant } A} + \sum_{q \in R(p)} \log \frac{P(h_q^{l'} | h_p^l = s)}{P(h_q^{l'})} = \\ A + & \sum_{t=1, \dots, K^{l'}} \left| \left\{ q : h_q^{l'} = t, q \in R(p) \right\} \right| \log \frac{P(h_q^{l'} = t | h_p^l = s, q \in R(p))}{P(h_q^{l'} = t)}. \end{aligned} \quad (2.19)$$

Denote by  $Q_t^{l'}(I, p) = \left| \left\{ q : h_q^{l'} = t, q \in R(p) \right\} \right|$  the number of times visual word  $t$  appears in the receptive field of location  $p$ . We look for a subset of words  $T \subset \{1, \dots, K^{l'}\}$ , which contribute the most to the likelihood of  $h_p^l = s$ . Thus, the problem we solve is

$$\max_{\substack{T \\ |T|=Z}} \left\{ \sum_{t \in T} Q_t^{l'}(I, p) \log \frac{P(h_q^{l'} = t | h_p^l = s, q \in R(p))}{P(h_q^{l'} = t)} \right\}. \quad (2.20)$$

The solution is obtained by choosing the first  $Z$  words for which the score

$$S^l(I, s, t) = Q_t^{l'}(I, p) \log \frac{P(h_q^{l'} = t | h_p^l = s, q \in R(p))}{P(h_q^{l'} = t)} \quad (2.21)$$

is the highest. Intuitively, the score of visual word  $t$  is the product of two terms,  $Q_t^{l'}(I, p)$ , which measures the word frequency in the receptive field, and  $\log \frac{P(h_q^{l'} = t | h_p^l = s, q \in R(p))}{P(h_q^{l'} = t)}$  which measures how likely it is to see word  $t$  in the receptive field compared to seeing it in general. To compute the probabilities in the log term of the score, we use the estimations  $\hat{P}(h^l = k)$  and  $\hat{P}(h^l = k' | h^l = k)$  given by Eqs. 2.14 and 2.15, respectively.

### 2.3.3.2 Explaining Multiple Words and Images

The optimization problem presented in Eq. 2.20 can be extended to multiple visual words in multiple images using column position and image independence assumptions. Assume a set of validation images  $\Omega$  is being analyzed, and a set of words  $S \subset \{1, \dots, K^l\}$  from layer  $l$  has to be explained by lower layer words for these images. We would like to maximize the likelihood of the set of all column activities  $\{h_p^l(I_n) : h_p^l \in S, I_n \in \Omega\}$ , in which a word from  $S$  appears. Assuming column position independence, this likelihood decomposes into terms similar to Eq. 2.18:

$$\begin{aligned} \log P \left( \left\{ h_p^l(I_n) : h_p^l(I_n) \in S, I_n \in \Omega \right\} \middle| \left\{ h_q^{l'}(I_n) : I_n \in \Omega \right\} \right) = \\ \sum_{n=1}^N \sum_{s \in S} \sum_{\{p: h_p^l(I_n)=s\}} \log P \left( h_p^l(I_n) \middle| h_q^{l'}(I_n), q \in R(p) \right). \end{aligned} \quad (2.22)$$

---

**Algorithm 1** Inference graph building

---

**Input:** CNN  $CN$ , a set  $\Omega$  of images predicted by  $CN$  to class  $m$ , network model

$\{\Theta^l, \hat{P}(h^l = k), \hat{P}(h^l = k|h^{l'} = k')\}_{l=1, k=1, k'=1}^{L, K^l, K^{l'}}$   
 $Z$  - number of allowed nodes per layer.

**Output:** An inference graph  $G = (N, E)$ , where  $N$  and  $E$  hold clusters (nodes) and their weighted connections (edges) in the graph, respectively.

**Initialization:** Push  $\Omega$  through the network model to get  $\{Q_{t,s}^l(\Omega)\}_{l=1}^{L-1}$  (Eq. 2.24) and clusters  $\{C_i^l\}_{l=1, i=1}^{L, K^l}$ .  
Set  $S = \{m\}$ ,  $N = C_m^L$ , and  $E = \emptyset$ .

For  $l = L - 1, \dots, 1$

For  $t = 1, \dots, K^l$ , compute  $S^l(\Omega, S, t)$  (Eq. 2.25)

Choose  $z_1^l, \dots, z_Z^l$  to be the  $Z$  clusters indices with the largest scores  $S^l(\Omega, S, t)$

Set  $S = \{z_1^l, \dots, z_Z^l\}$  and  $e_{i,j}^l = S^l(\Omega, z_i^{l+1}, z_j^l)$ ,  $\forall i, j = 1, \dots, Z$

Set  $N = N \cup \{C_{z_i^l}^l\}_{i=1}^Z$  and  $E = E \cup \{e_{i,j}^l\}_{i=1, j=1}^{Z, Z}$

---

Repeating the derivation also given in Eqs. 2.18, 2.3.3.1, and 2.20 for this expression, we get a similar optimization problem,

$$\max_{|T|=Z} \left\{ \sum_{t \in T} \sum_{s \in S} Q_{t,s}^{l'}(\Omega) \log \frac{P(h_q^{l'} = t | h_p^l = s, q \in R(p))}{P(h_q^{l'} = t)} \right\}, \quad (2.23)$$

where  $Q_{t,s}^{l'}(\Omega)$  is the aggregation of  $Q_t^{l'}(I, p)$  over multiple positions and images

$$Q_{t,s}^{l'}(\Omega) = \sum_{n=1}^N \sum_{\{p: h_p^l(I_n) = s\}} Q_t^{l'}(I_n, p). \quad (2.24)$$

That is,  $Q_{t,s}^{l'}(\Omega)$  is the number of occurrences of word  $s$  with word  $t$  in its receptive field in all the images in  $\Omega$ . The solution is given by choosing the  $Z$  words in layer  $l'$  for which the score

$$S^{l'}(\Omega, S, t) = \sum_{s \in S} Q_{t,s}^{l'}(\Omega) \log \frac{P(h_q^{l'} = t | h_p^l = s, q \in R(p))}{P(h_q^{l'} = t)} \quad (2.25)$$

is maximized. Like the score in Eq. 2.21, the contribution of word  $t$  to the explanation of a single word  $s$  is a product of its frequency in the relevant receptive fields and its discriminative value term.

The inference graph is generated by going over the layers backwards, from the top layer, for which the decision has to be explained, and downwards towards the input layer, selecting the explaining nodes using the score of Eq. 2.25. See Algorithm 1 for details.

## 2.3.4 Visualization Techniques

We consider visualization at several levels starting from simple visual word to path inference, and relating to the two types of networks.

### 2.3.4.1 Simple Cluster

We visualize a cluster  $C_k^l$  by showing the  $m$  examples ( $m = 6$ ) with the highest  $P(h_p^l(I) = k)$  across  $(I, p) \in S_v$ . For FC layers, for each representative example, the full image is shown. For convolutional layers, visual word examples are typically sub-regions of the input image (if the receptive field does not cover the image entirely). For receptive fields larger than 25% of the image size, the visual word occurrence is visualized by drawing the relevant image with a red rectangle around the relevant receptive field (see Fig. 2.1). When the receptive field is smaller, only the region of the receptive field is shown instead of the entire image.

### 2.3.4.2 Cluster as a Decision Junction

In MLP networks, the entire layer activity is assigned to a single cluster. We consider such a cluster as a “decision junction”, where a decision regarding the consecutive layer cluster is made. For the visualization of such a decision, the activity vectors assigned to  $C_k^l$  are labeled according to their cluster index in the *consecutive layer*, thereby forming sub-clusters. We use linear discriminant analysis (LDA) [35] to find a two-dimensional projection of the activities that maximizes the separation of the examples with respect to their sub-cluster labels. To understand the semantics of the sub-clusters, we draw the three *most typical* representative examples from each sub-cluster near the sub-cluster centroid. We define *most typical* examples as the three minimal  $l_2$  distance images to the sub-cluster center (see Fig 2.4).

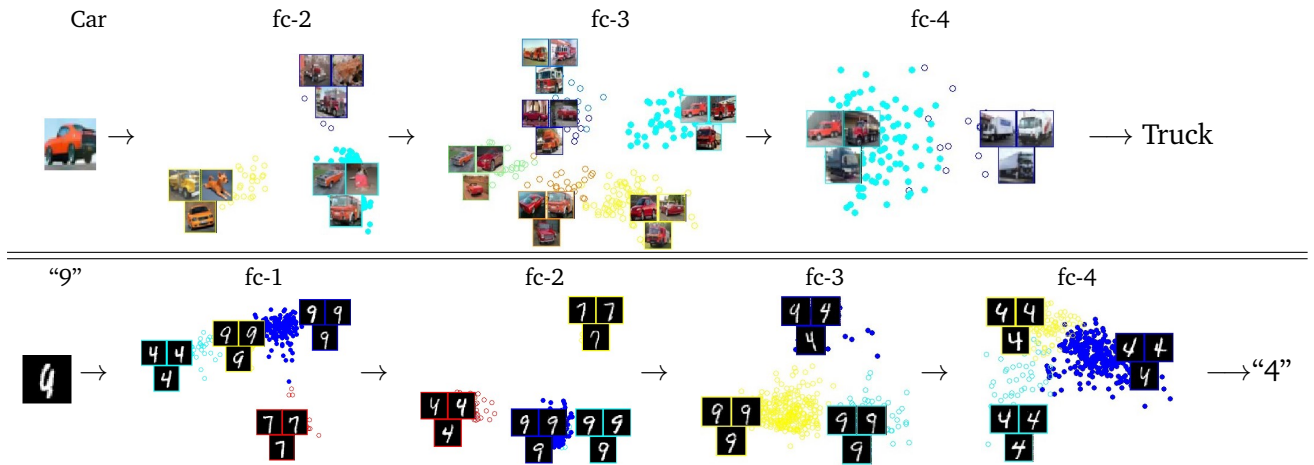
### 2.3.4.3 Inference Graphs

For MLP networks, the inference path of a specific example  $I$  contains a single visual word at each layer. It can be defined by the maximum a-posterior (MAP) cluster sequence, i.e., the sequence  $H = (h^1, \dots, h^L)$  satisfying

$$\max_{h^1, \dots, h^L} \log P(h^1, \dots, h^L | X(I)). \quad (2.26)$$

$H$  can be found using the Viterbi algorithm [36]. The path nodes are visualized using the decision junction technique of Section 2.3.4.2.

For CNNs, multiple spatial words are active at each layer. We use the technique explained in Section 2.3.3.2 (Algorithm 1) to generate inference graphs highlighting the main active words. Such graphs can be built for an entire class by choosing the input  $\Omega$  of Algorithm 1 to be the set of all images predicted to the class, or for a single example. In the visualization of such graphs,



**Figure. 2.4: Two inference paths in MLP.** We show the main decision junctions for an example six-layer MLP network. The analyzed examples are presented on the left. Cluster representatives are chosen using the  $l_2$  metric. In each decision junction, the points of the sub-cluster chosen by the example are marked with full circles in cyan (top) blue (bottom) **Top:** The path of a misclassified CIFAR10 car example is shown. The main decision points occurred in layers fc-2 – fc-4, with the critical decision made in layer fc-3 where the abnormal appearance of the example misled the network to consider the car as a truck. **Bottom:** The path of a misclassified MNist "9" digit example is shown. The input image traversing main decision clusters through layers fc-1 – fc-4, where the main flawed decision is made in layer fc-3, where the open head of the "9" image misleads the network to consider as "4" digit.

each visual word is displayed using its  $m$  most representative spatial examples in the validation set. Connections between words are characterized by their contribution to the log-ratio component in the maximum likelihood score (Eq. 2.25, right term).

For inference graphs of a specific image  $I$ , node visualization also shows all the spatial locations in  $I$  belong to the corresponding visual word.

### 2.3.5 Method Limitations and Assumptions

The probabilistic model presented, requires  $2D^l K^l + K^l$  additional parameters for each new modeled layer. As we use a GMM for activations modeling, a single forward pass calculates the conditional probability of each spatial location over the  $K^l$  modeled Gaussians. This yields an activation tensor size  $W^l \times H^l \times K^l$ . Since  $K^l$  typically grows linearly with the number of classes whose inference is modeled, the model is less suitable for modeling many classes simultaneously. Furthermore, in backward steps, the derivative of a Gaussian pdf is calculated with some additional computational cost. However, if training is done post-hoc, typically only  $\sim 20$  epochs are required until model convergence.

In order to enable a feasible probabilistic model, some assumptions were made in SIGN. The most prominent is the assumption of independence of consecutive layers and of spatial columns activations. This is not valid since the receptive fields of close activations overlap, and they are



propagated from the same regions in lower layers. However, such simplifications are necessary for modeling each layer separately using a GMM, as well as calculating the gain score of a single visual word (Eq. 2.18). A second simplifying assumption is the use of a diagonal covariance matrix in the GMM components, required to avoid an  $O(KD^2)$  parameter complexity. This amounts to an independence assumption between different maps within the same layer.

While these simplifying assumptions have been made, the resulting model enables statistical inference, and provides a useful XAI framework.

## 2.4 Results

### 2.4.1 Implementation Details

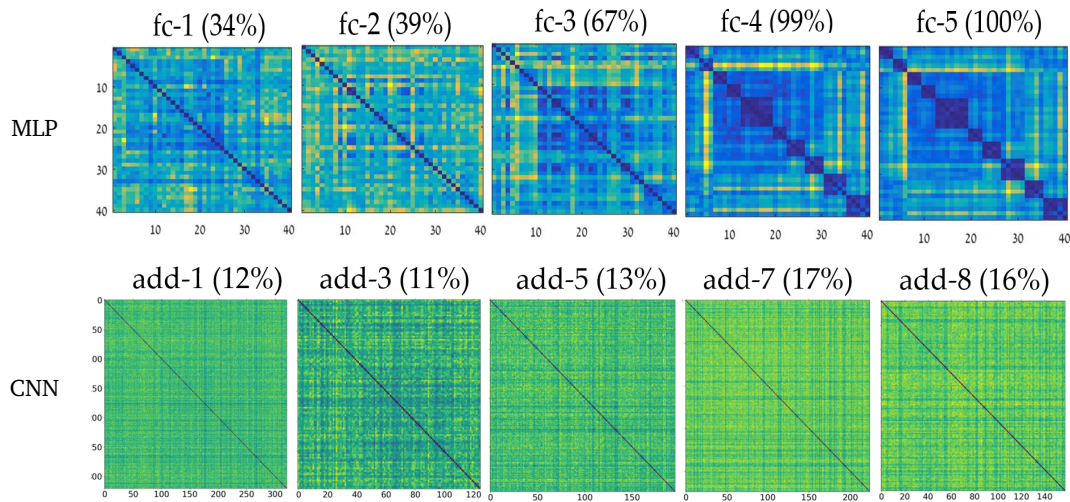
The HMM for MLP formalism was tested by training fully connected networks on the MNIST [37] and CIFAR10 [38] datasets, containing 10 classes each. The networks included six layers with the first five containing 1,000 neurons each (the last layer has 10 neurons according to the number of classes). Based on a preliminary evaluation, the number of visual words  $K^l$  was set at 40 for all layers.

CNN models included ResNet20 [39] trained on CIFAR10, and VGG-16 [8] and ResNet50 trained on the ILSVRC 2012 dataset [13]. For ResNet20, the output of all add-layers after each skip connection were modeled, as these outputs are expected to contain aggregated information. For ResNet50, there are 16 add-layers and the output of add-layers 3, 7, 13, and 16 were modeled. For VGG-16, the first convolutional layers at each block were modeled (block1\_conv1,..., block5\_conv1). The numbers of visual words were set at 60, 100, 200, 450, and 1,500 for layers 1–5, respectively, according to the GPU memory limitation.

In all experiments and modeled layers, the GMM's mean parameters were initialized using  $K^l$  randomly selected examples. The variance parameters were initialized as the variances computed from 1,000 random examples. Prior probabilities were uniformly initialized to be  $\frac{1}{K^l}$ .

### 2.4.2 Inference Modeling in MLP networks

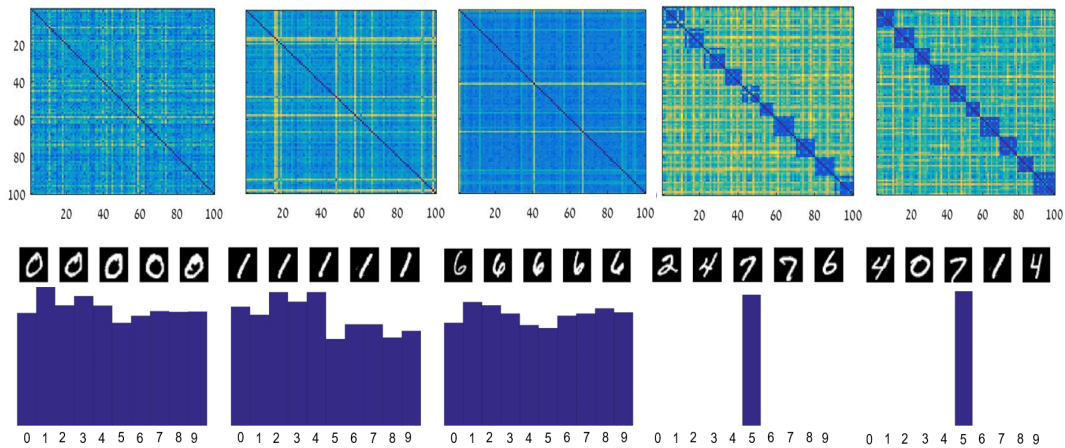
**Sequential path.** Fig 2.4 depicts how inference paths, drawn as sequences of decision nodes, are useful for error diagnosis. In Fig. 2.4 (top), a path of an erroneous "car" example in the CIFAR10 network is partially presented. The sub-clusters containing the example are marked with full cyan circles. While layers fc-2 and fc-4 primarily make decisions based on color, the wrong decision leading to the mis-classification of the car as "truck" is made at layer fc-3. The example's cluster in layer fc-3 contains six sub-clusters, leading to car and truck clusters in the consecutive layer. At



**Figure. 2.5: Cluster development across layers.** Cluster similarity matrices for increasing layer indices in MLP and CNN networks trained for CIFAR10 classification. In each matrix, Euclidean distances between cluster centers for clusters of a single layer are shown. Clusters are ordered by their dominant class (clusters with dominant class 1, followed by all clusters with dominant class 2, etc.). Layer index and average cluster purity (%) are shown above each matrix. **Top: MLP activity.** Growing similarity between clusters with the same dominant class can be seen by the appearance of a block diagonal structure, from layer 3 onwards. This indicates convergence toward a single class-specific representation as layers progress. **Bottom: CNN activity.** Clusters of add-layers of ResNet blocks 1, 3, 5, 7 and 8 are presented. Here, no similarity is formed with layer growth; each class is represented by multiple localized visual words, which have unrelated activity patterns.

this point, the example was wrongly associated with the sub-cluster representing "truck" due to its exceptional rear appearance, resembling the appearance of a truck front. From this point onwards, the path is associated with "truck" clusters, up until the classification layer. In Fig. 2.4 (bottom), a path of an erroneous "nine" example in the MNIST network is partially presented with full blue clusters. Correct network decisions are made in layers fc-1 and fc-2, where the network associates the example with primary "nine" sub-clusters. The wrong decision of the network is made in layer fc-3, where it decided to "send" the example to a "four" cluster in layer fc-4, continuing with this pattern up until the classification layer.

**Cluster similarity development across layers.** Progressing through FC layers, activity clusters tend to become more class oriented, i.e., dominated by examples from a single class. Furthermore, these clusters become increasingly similar with layer index progression, indicating convergence of class examples toward a single class-specific representation. For each cluster, we define its class naturally as the class whose examples are the most frequent among the cluster examples, and the class dominance index is the percentage of dominant class examples. In Fig. 2.5 (top) cluster purity and inter-cluster distances are shown for the CIFAR10 FC network layers. The similarity of clusters representing the same class gradually increases in layers 3-5, as evident from the emerging block structure. It can be observed that a significant portion of the training occurs in the middle layer of the network, specifically in the transition between the third representation (last layer of the first half) and the fourth (first in the second half). This phenomenon was enhanced in the case of severe overfit discussed below.

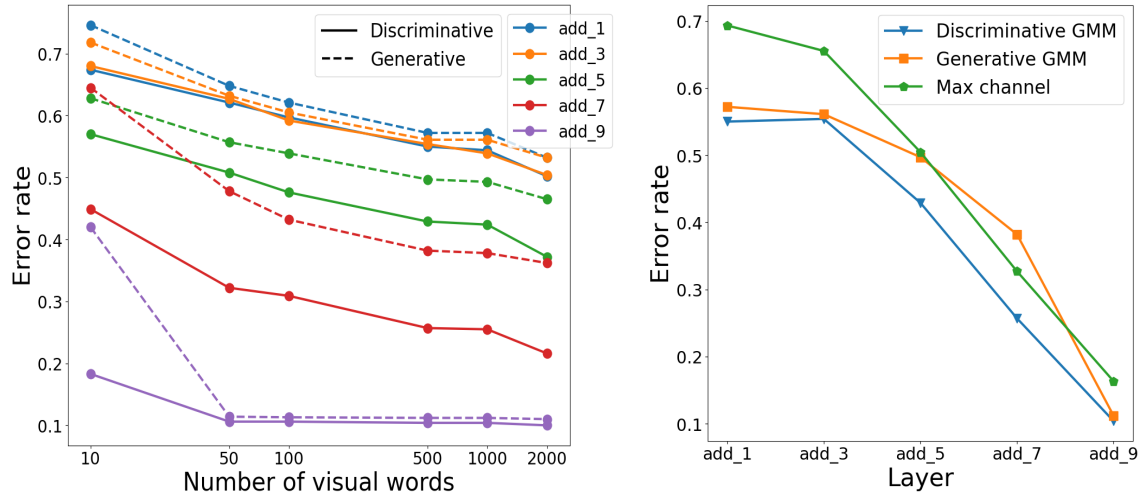


**Figure. 2.6: Cluster development for an extreme overfit case.** Cluster distances and average purity for a network trained with random labels are shown. **Top:** Cluster similarity matrices for layers 1-5. **Bottom:** Typical clusters at these layers. For each cluster the (pseudo) label histogram is shown, as well as some representative cluster images. An abrupt transition from input-dominated to output-dominated representation occurs in the transformation between the third and fourth layers.

**Overfitting capacity in a single layer.** It is known that networks can be trained to an extreme overfit condition by using randomly generated labels in training [14]. The resulting classifier has a training error of 0, meaning that it had successfully memorized a mapping between all training images to their pseudo labels, but its generalization error is of chance level. We utilize our model to understand how this memorization mapping is formed. A six FC-layer network was trained on the 60,000 examples of the MNIST data with random labels, until reaching a zero training error. In Fig. 2.6, the similarity matrices between cluster centers for layers 1–5 are shown (top), with typical clusters at each layer (bottom), presented using their label histograms and representative images. Surprisingly, one can observe a concentration of the network overfit behavior in a single layer transformation between the third and fourth layers. In layers 1 – 3, clusters are input related. They contain images with similar appearance, hence, with similar true labels (and uniformly distributed pseudo labels). These clusters are not close in the Euclidean sense. In Layer 4, there is a sharp transition to clusters which are completely (pseudo) label dominated, as indicated by their class purity (histogram) and block structure in the similarity matrix. The fact that the memorization of the full data (60,000 instances) concentrates almost completely at a single transformation in the middle of the network (between layers 3 and 4) is a novel unexpected observation, for which we do not currently have a good explanation.

### 2.4.3 Inference Modeling in CNNs

**Loss and dictionary sizes.** The discriminative quality of a visual dictionary can be quantified by using it to form word histograms, then checking the error of a linear classifier on this representation (a bag-of-words methodology). Fig. 2.7(Left) shows the errors obtained for dictionaries trained with losses  $\mathcal{L}_G$  (2.16) and  $\mathcal{L}_D$  (2.17) on intermediate convolutional layers of a ResNet20 network on CIFAR10. The graphs show error rates as a function of the layer index and dictionary size. It can



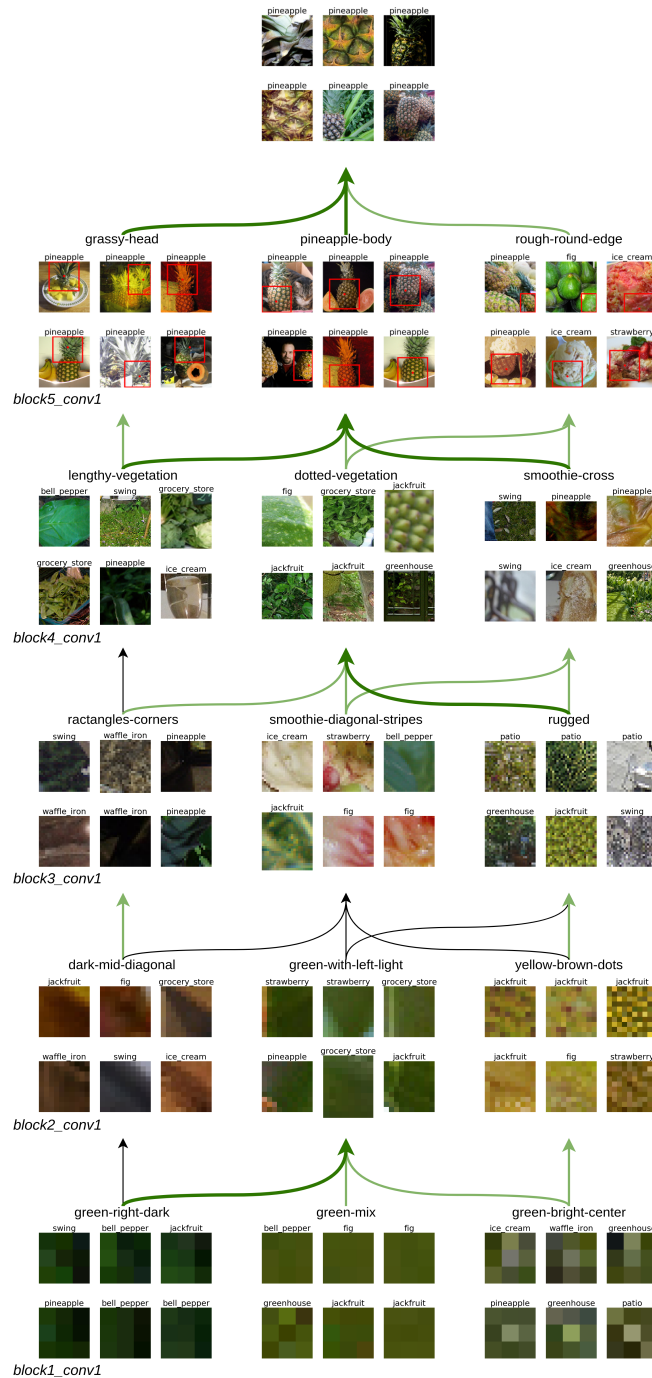
**Figure. 2.7: Classification accuracy analysis as a function of dictionary size and layer depth.** Error rates of linear classifiers, based on cluster/visual word histograms are shown. All experiments conducted on five ResNet20 conv-layers, trained on CIFAR10. **Left:** Error rates based on SIGN generative and discriminative loss functions (Eqs. 2.16 and 2.17), as a function of dictionary size. **Right:** Error rates of SIGN clustering method and plain clustering based on the maximal active channel (see text for explanation).

be seen that errors decrease with the layer index (from 1 to 9), reaching the original network error of 0.088. As expected, the discriminative loss  $\mathcal{L}_D$ , whose minimization directly decreases the error, leads to a higher accuracy than the generatively-optimized loss. Therefore, all models presented below were trained with the  $\mathcal{L}_D$  loss. In addition, the error rate decreases monotonically with the dictionary size for all layers, but the gain from dictionary sizes larger than 500 is minor in most cases.

To validate SIGN clustering scheme, the suggested representation’s accuracy is compared to a baseline clustering method, in which clusters correspond to channels of the output tensor. In the baseline method, each column activation  $x_p^l$  is assigned to a cluster based on its maximal activity channel. The cluster histogram is then fed to a linear classifier in the same procedure as mentioned above. Error rates of SIGN representations are shown compared to those of the max channel clustering representations across the modeled layers in Fig 2.7(Right). For SIGN method, we present the error rate based on 500 visual words in all modeled layers, both for discriminative and generative losses. For max channel clustering method, we used the discriminative loss (i.e., cross-entropy based on clusters bag-of-words). The discriminative GMM loss produced the lowest error rate across all layers, with significant margin over the plain channel-wise clustering. Relative improvement from the baseline technique ranges from 20% in lower layers, up to 45% in the highest layer add\_9.

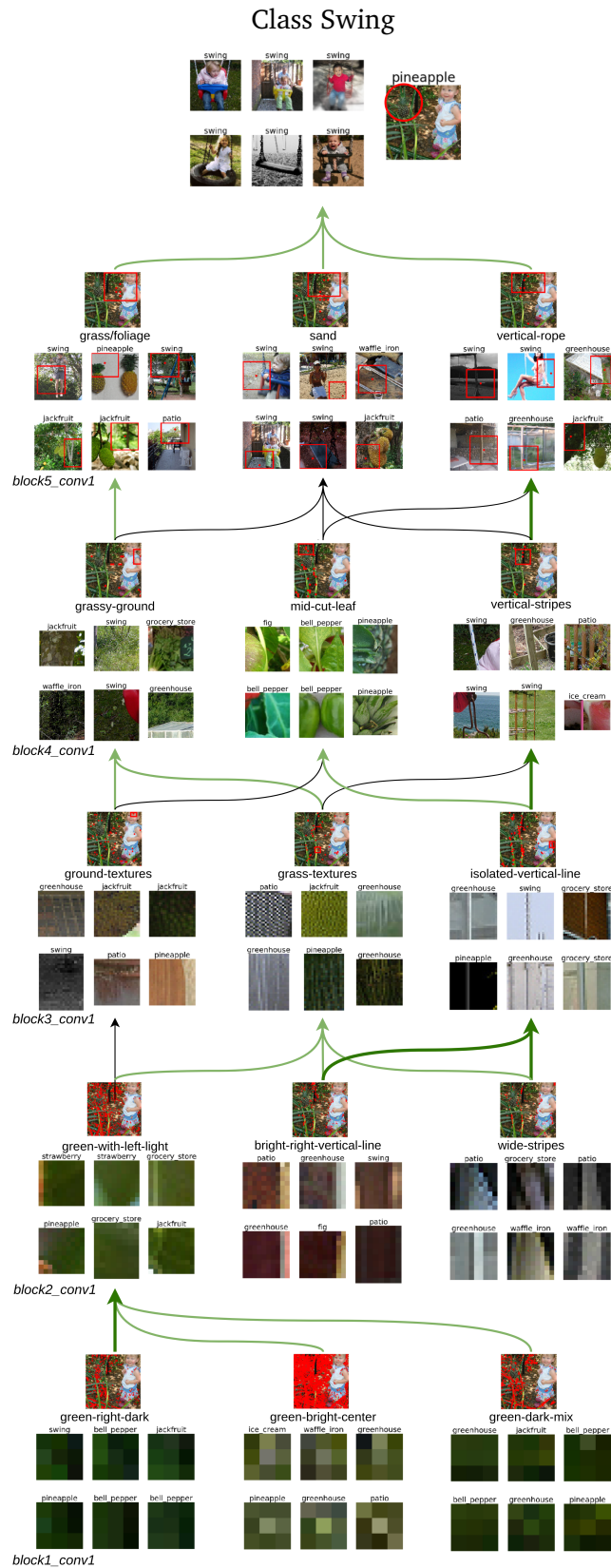
**Cluster development across layers.** Fig. 2.5 (bottom) shows the cluster distance matrix for several convolutional layers of a ResNet20 network trained on CIFAR10. Unlike in MLPs, learned clusters represent activity at specific spatial locations. Even though receptive fields of clusters from advance

## Class Pineapple



**Figure. 2.8: Pineapple inference graph.** The graph is generated by training a model on the “pineapple” class and its neighboring classes. The top node is a visual word of the output layer, representing the predicted class “pineapple”. The lower levels in the graph show the three most influential words in preceding modeled layers (block5\_conv1, ..., block1\_conv1). Visual words are manifested by the six representative examples for which  $P(h^l = k|x_p^l)$  is the highest. For modeled layer block5\_conv1, examples are presented by showing the example image with a rectangle highlighting the receptive field of the word’s location. For lower layers, the receptive field patches themselves are shown. Images are annotated by their true label. Arrows are shown for the two most significant connections for each lower visual word. When the log-ratio term (right element in Eq. 2.25) is positive, it is colored (1) black:  $0 < \text{log-ratio} < 1$ , (2) light green:  $1 \leq \text{log-ratio} < 2$ , (3) mild green:  $2 \leq \text{log-ratio} < 3$ , or (4) dark green:  $3 \leq \text{log-ratio}$ . In addition, a tag above each visual word was added by the authors for convenience. The figure is best inspected by zooming in on clusters of interest.





**Figure. 2.9:** An image inference graph of an erroneous image. An image inference graph for a pineapple image wrongly classified to the class "swing". The model is trained using "pineapple" and its neighboring classes (same as in Fig. 2.8), where the neighbor class "swing" is included. The graph is generated by applying the node selection algorithm (Section 2.3.3) to a set  $\Omega$  containing this single erroneous image. The analyzed image is shown on the top of each cluster node, with red dots marking spatial locations assigned to the cluster. In the top node, the pineapple object is marked in red circle for clarification.

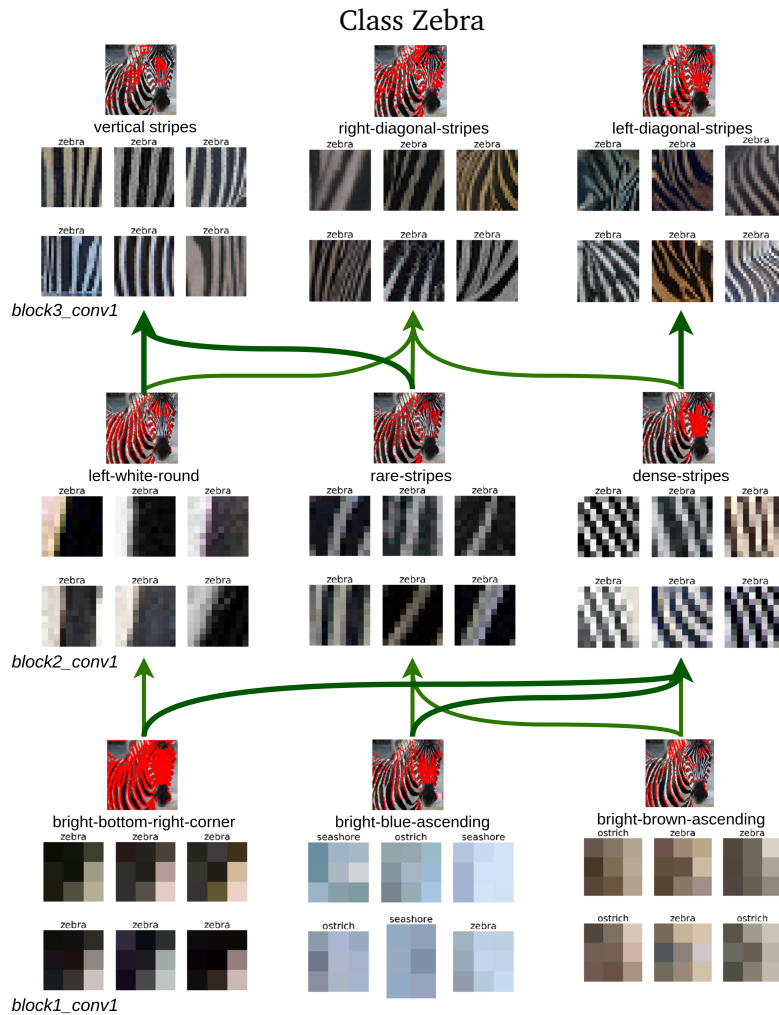
layers cover the entire input space, there is no apparent similarity between clusters with the same dominant class. This indicates that activity columns of advanced layers remain local, an observation also supported by the inference graph visualizations shown below. While clusters do become more class specific, they are less specific than in the MLP network, and the final classification is based on several class-specific words appearing simultaneously in different image regions.

**Class inference graph.** An example of a class inference graph for the class “pineapple” in VGG-16 is presented in Fig. 2.8. This graph is generated by training a clustering model on the “pineapple” and related classes (see Section 2.3.2), then applying the node selection algorithm (Section 2.3.3) to the set  $\Omega$  of pineapple images included in the validation set. The graph shows that the most influential words in the top convolution layer can be roughly characterized as “grassy-head”, “pineapple-body”, and “rough-round-edge”. The origins of these words can be traced back to lower layers. For example, “grassy-head” is composed of word capturing mostly “lengthy-vegetation” in the layer below. The “pineapple-body” is composed of words contain “vegetation” types and “cross” textures (block4\_conv1) which are in turn generated from words describing mostly green and yellow textures and shapes (block3\_conv1).

**Image inference graphs.** Fig. 2.9 shows an image inference graph for a pineapple image wrongly classified to the “swing” class. Using the inference graph, we can analyze the dominant (representative) visual words that led to this erroneous classification:

- block5\_conv1 (top layer): The visual words connected directly to “swing” class, hence, causing the error, can be characterized as “grass/foilage-texture”, “sand”, and “vertical-rope”. Such words are indeed statistically related to swing presence in images, and many map locations in the inspected image are assigned to them.
- layers block4\_conv1 and block3\_conv1: The “vertical-rope” (block5\_conv1) originates from a similar visual word, “vertical-stripe”, of layer block4\_conv1, and this in turn depends strongly on the “isolated-vertical-line” word in layer block3\_conv1. The foliage word (block5\_conv1) mainly originates from the “grassy-ground” word in layer block4\_conv1, which in turn heavily depends on the two “ground-structure” and “grass-structure” words in layer block5\_conv1.
- layer block2\_conv1: The main explanatory words are green and bright vertical edges and lines, which are combined to construct the “isolated-vertical-line” and “grass-structure” words in layer block3\_conv1.

In Fig. 2.10, we show part of the inference graph for a successfully classified zebra image, focusing on the bottom three layers. The gradual development of discriminative stripe-based features can be seen. Visual words in block3\_conv1 (top layer) are each characterized by a single orientation: vertical (left), leaning to the right (middle), or leaning to the left (right). These words are less sensitive to spatial frequency. They abstract over the spatial frequency by combining words from layer block2\_conv1 that mostly differ w.r.t their line spatial frequency and edge patterns. In layer

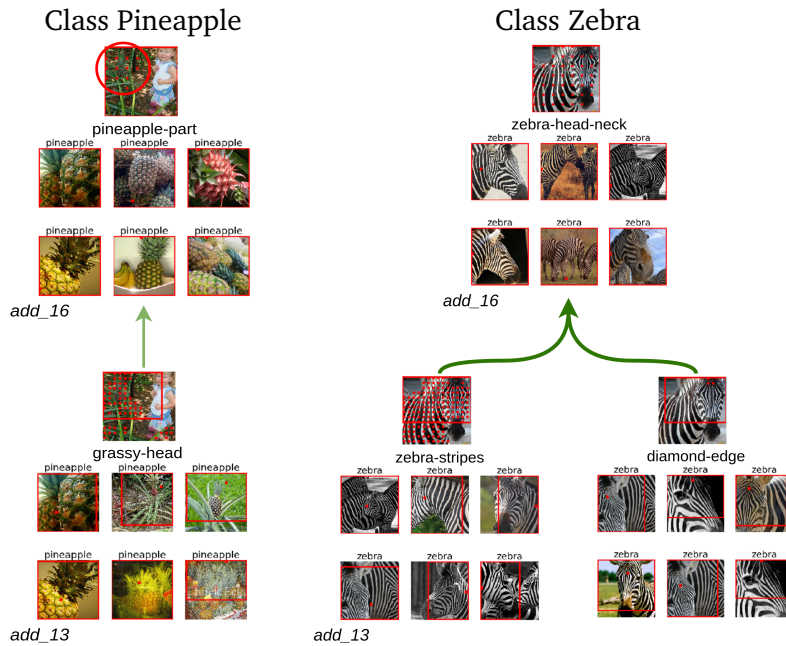


**Figure. 2.10:** Partial image inference graph of a correctly classified zebra image. The figure presents only the main contributing visual words from the three bottom modelled layers.

block1\_conv1, we encounter the edge feature patterns composed to create the words of layer block2\_conv1.

In Fig. 2.11, we show partial inference graphs for ResNet-50. On the left, upper nodes from the inference graph of “pineapple” image from Fig. 2.9 are shown. Unlike VGG-16, ResNet50 successfully classifies this image. As can be seen in layer add\_16, ResNet50 successfully detects the pineapple location in the image (marked in green circle), where both visual words presented contain strong “pineapple” features. On the right, we show upper nodes of the ResNet-50 inference graph for the “zebra” image from Fig. 2.10, successfully classified also by ResNet-50. The top word shown usually captures the zebra’s head, with its receptive field center located on the neck. This “zebra-head-neck” visual word is formed from the bottom-left word representing a near-head stripes texture, and the bottom-right word which captures a diamond shape located between the zebra’s eyes. Both words are discriminative, as indicated by their log-odds score (higher than 2).



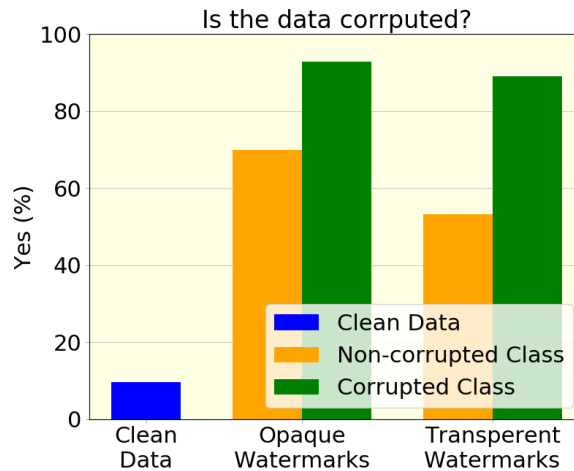


**Figure. 2.11: Sub-graph inference for ResNet50.** **Left:** The pineapple image wrongly classified to class "swing" in VGG-16, presented in Fig. 2.9, is correctly classified to "pineapple" class in ResNet50 (the pineapple object is marked in green circle). The sub-graph presents the visual word from the top layer (add\_16) connected to a visual word from the lower layer (add\_13). **Right:** The zebra image from Fig. 2.10, correctly classified in VGG-16, is correctly classified in ResNet50 as well. The sub-graph includes a visual word from the top layer (add\_16) aggregated from two visual words from the lower layer (add\_13).

#### 2.4.4 Biased Data Experiment

A user experiment with human subjects was carried in order to test whether SIGN can effectively enable detection of artifacts in the data. Specifically, SIGN was tested if it enables a user detection of class-specific artifacts using class inference graphs. That is, we injected class-specific biases in the data where all images of a certain class were identified with a unique reoccurring artifact. For the experiment, we trained five ResNet20 models. One was trained to classify CIFAR10 (baseline), and four others were trained with CIFAR10 versions where images were corrupted with watermarks. In each experiment with corrupted data, a single purple English letter was placed at a random position in images of 2 classes of the 10. In two of the experiments, images contained clearly visible opaque watermarks, and in the remaining two experiments images contained hard-to-detect transparent ( $\alpha = 0.5$ ) watermarks. Class inference graphs were formed for a couple of classes in each experiment, and users were asked to determine using the graph whether the data of the corresponding experiment is clean or corrupted. Since SIGN finds features associated with a certain class, we expected the distinctive features to be found by the graph.

A total of 60 subjects of different ages and backgrounds volunteered to participate in the experiment. Each participant was presented with 16 inference graphs. Among the 16 graphs, 4 graphs were produced from clean data, 6 from a model trained with opaque watermarks, and 6 with transparent



**Figure. 2.12: Biased data experiment results.** Users were asked to determine whether inference graphs shown to them contain corrupted data. The graphs shown were produced using ResNet20 trained either with clean data (blue), or with opaque/transparent watermarks induced in 2 classes of CIFAR10. For the datasets with watermarks, users were shown inference graphs of corrupted classes (green) and non-corrupted classes (orange). Users easily detected corruptions in class inference graph of classes stained with watermarks. When shown inference graphs of non-corrupted classes from the same corrupted dataset, the user accuracy is lower, especially with transparent watermarks.

watermarks. Among the 12 graphs corrupted with watermarks, we presented 6 graphs from the corrupted classes and 6 graphs from the non-corrupted classes.

Experiment results are shown in Fig 2.12. The accuracy of users is shown for each bias scenario. Participants classified clean data correctly with 90% accuracy. For graphs with opaque watermarks, participants found corruption with 93% accuracy when presented with class inference graphs of the corrupted class and detect corruption with 70% when shown inference graphs of the non-corrupted class. This difference intensifies with transparent watermarks, where watermarks are less visible. 90% of the participants correctly classified the data as corrupted when shown the corrupted class inference graphs and 53% when shown class inference graphs of non-corrupted classes.

When producing class inference graphs of corrupted classes (Fig 2.13), the SIGN model finds clear watermark-related visual words in all modeled layers as a strong feature of the class. It can be observed that the red center point of the receptive field patch is always located in proximity to the watermark. In upper layers `add_6` and `add_8`, it can be seen that without the red center point, pointing at the watermark location, it is difficult to detect these type of corruption. When producing graphs of non-corrupted classes, watermark-related visual words do not appear in lower layers, and the users find it harder to detect watermarks.

For the suggested experiment, where images of an entire class are contaminated by a unique watermark, we can expect that attribution methods will detect the watermark. Indeed, as shown in Figure 2.14, Grad-CAM [9] deployed on the last convolutional layer of ResNet20 highlights the watermarks in stained classes. However, Grad-CAM visualization only analyzes the impact of the



**Figure. 2.13: Corrupted data debugging with SIGN.** Class inference graph of "truck" class from ResNet20, trained on biased CIFAR10. The data was corrupted by inducing 2 classes with transparent watermarks ( $\alpha = 0.5$ ) at random places. The above graph shows the "truck" class inference graph in an experiment where classes "truck" and "cat" were corrupted with purple letters "T" and "A" respectively. It is recommended to zoom in for better inspection.



**Figure. 2.14: Attention maps demonstration on biased data.** (a) Examples of CIFAR10 images corrupted with transparent watermarks along with their (b) corresponding Grad-CAM [9] heatmaps.

final convolutional layer on the decision. The SIGN method provides deeper analysis by enabling understanding of the full inference process through the hidden layers based on the statistical analysis of the full training population. Such analysis enables understanding semantics of error sources (Figure 2.9), feature development (Figure 2.10), and even identification of layers responsible for overfitting (Figure 2.6).

## 2.5 Conclusions

In this paper, we introduced SIGN, a new approach for interpreting hidden layers activity of deep neural networks based on learning activity cluster dictionaries and transition probabilities between clusters of consecutive modeled layers. We formalized a maximum-likelihood criterion for mining explanatory clusters, and an algorithm for the construction of inference graphs with manageable sizes. Inference graphs can be constructed for entire classes, to understand the general network reasoning for this class, or for specific images for which error analysis may specifically be sought.

The tools developed here can be used to verify the soundness of the network reasoning and to better understand the network's hidden mechanisms, or conversely, reveal weaknesses and main error causes. Network debugging is currently a difficult and daunting task, and we believe the suggested tools may be a useful component in a developer's debugging toolbox. Beyond its utility in network interpretation and debugging, the suggested approach and tools revealed several surprising network behavior patterns, such as the extreme locality of activity columns in top CNN layers, and the concentration of memorization in a single intermediate middle layer in fully connected networks.

Several interesting avenues are open for future work. One such avenue may be re-training more explainable networks by enforcing, during training, only activity of clusters and connections with high explanatory value. Another direction is to use the models in a task-transfer scenario. Network refinement for a new task may be constrained to use only relevant activation clusters, as determined by a human observer. In this way, a human may use the suggested tool to define a relevant prior

for the new task. Finally, we may try to design explanatory capability that goes beyond statistical analysis and maximum-likelihood justification, and into causal analysis based on intervention.

# Detection of Overlapping Ultrasonic Echoes with Deep Neural Networks

## Publications:

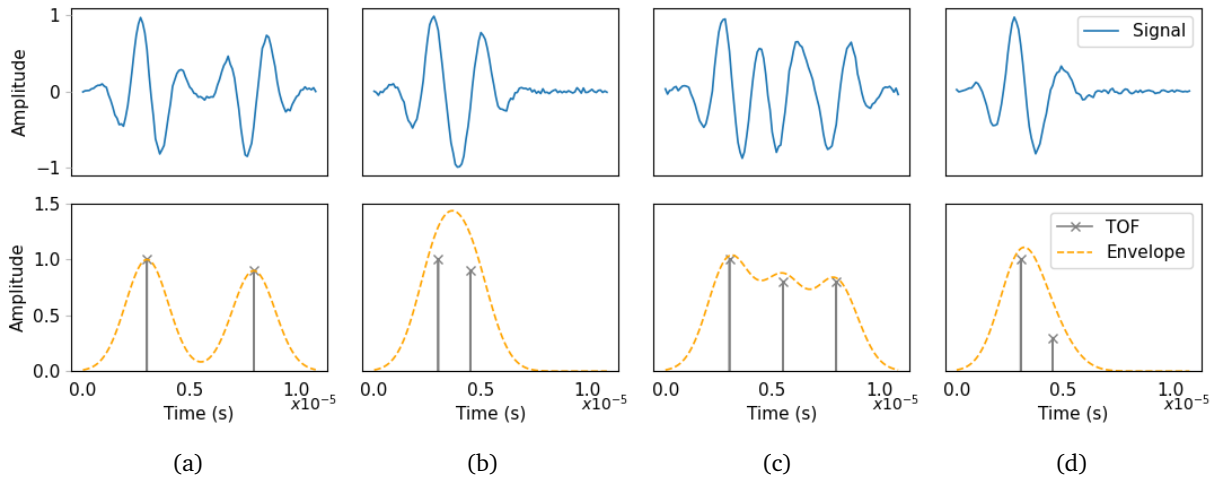
- **Shpigler, A.**, Mor, E., & Bar-Hillel, A. (2022). Detection of overlapping ultrasonic echoes with deep neural networks. *Ultrasonics*, 119, 106598.

## Abstract:

Ultrasonic Pulse-Echo techniques have a significant role in monitoring the integrity of layered structures and adhesive joints along their service life. However, when acoustically measuring thin layers, the resulting echoes from two successive interfaces overlap in time, limiting the resolution that can be resolved using conventional pulse-echo techniques. Deep convolutional networks have arisen as a promising framework, providing state-of-the-art performance for various signal processing tasks. In this paper, we explore the applicability of deep networks for detection of overlapping ultrasonic echoes. The network is shown to outperform traditional algorithms in simulations for a significant range of echo overlaps, echo pattern variance and noise levels. In addition, experiments on two real adhesively bonded structures are conducted, demonstrating superiority of the network over traditional methods for layer thickness estimation.

## 3.1 Introduction

Ultrasonic pulse-echo methods have been widely used in the analysis of thickness and bond quality of layered materials in the Non-Destructive Testing (NDT) domain [40]. With these methods, an ultrasonic transducer sends an ultrasonic pulse and receives the reflected echoes from interfaces in the inspected sample. The arrival time of reflected echoes indicate the position of the interfaces, while amplitudes are used to evaluate the interface condition. However, the performance of these methods is limited for thick layers in which successive echoes do not overlap in time. Such overlap occurs when the time-of-flight difference (TOFD) in the layer is shorter than the pulse width, forming an axial resolution problem. In this case, simple derivation of individual echo parameters is not possible. An additional challenge often posed in this respect is reverberating waves from the inspected layers, forming successive overlaps.



**Figure 3.1:** Illustrations of ultrasonic overlapping echoes (top) along with their envelopes and TOFs (bottom). (a) Minor overlap, (b) major overlap, (c) successive overlaps, and (d) overlap with a low-amplitude echo.

Large overlaps between echoes create a difficult pattern recognition problem. First, it is difficult to understand how many echoes are involved. Second, it is difficult to determine their exact time-of-flight (TOF). An illustration of several difficult cases of overlap detection is shown in Fig. 3.1. One way to overcome this problem would be the use of high frequency, wideband transducers, which reduce the temporal pulse width and therefore enhance the resolution between successive echoes. However, due to increased attenuation, higher frequencies provide less penetration through the material layers.

Another difficulty arises when echoes are not identical, and their shape is not fully known a-priori. While the echoes are similar to a prototypical pulse originally sent by the transducer, they may have wide variance in their phase, center frequency, width and amplitude. These variations are caused by several physical reasons, including frequency depended attenuation and dispersion.

Numerous methods were suggested over the past several decades to solve this task. A simple and classical technique for solving overlapping echoes is by the amplitude spectrum method [41, 42], where a signal containing overlapping echoes is separated by the minima in the frequency-domain. This method is suitable for estimating the TOFD of a single layer embedded between two thick layers. However, it is not situated for separation of echoes caused by several unknown layers, and can also be sensitive to noise, especially when the resonance minima is at the high end of the bandwidth where the Signal-to-Noise Ratio (SNR) is low. Another approach is model-based signal decomposition by estimation of the echo parameters [43, 44]. While this approach is rather simple and does not rely on a reference signal, it can only converge to a local minimum, thus is highly depended on a good initialization. Moreover, this approach usually requires the number of echoes, composing the signal, to be known in advance.

Another popular approach is using sparse signal representation (SSR) methods, otherwise known as dictionary-based methods [45, 46, 47, 48]. These techniques have been widely used for

ultrasound echoes separation thanks to their efficiency in adaptive signal decomposition [40]. A major limitation of these techniques arises with growing variance in the echo patterns. To enable successful detection, the dictionary might require multiple values for each parameter, and a Cartesian product of the values used for multiple parameters. Dictionary size hence grows exponentially in the number of parameters, which leads to slow detection and higher false alarm rate.

Convolutional neural networks (CNNs) [49] have emerged as a promising direction for a variety of complex pattern recognition problems, and in recent years were adopted in the signal processing domain to solve sequence processing problems like text-to-speech [50], music synthesis [51, 52] and speech enhancement [53, 54, 55]. Contrary to the classical inverse methods which rely solely on a proximal physical model and simple priors, these methods are learned from collected data in which ground-truth reconstructed signals are known for measurement examples. They hence enable learning more complex data-dependent priors and predictions. In image recognition, convolution layers were shown to successfully model complex features and object parts [56]. Similarly, such layers can potentially model echo parts and overlapping echoes, and learn to reconstruct overlaps with different shapes.

Fundamentally, the basic building blocks in CNN architecture, convolutions, are well suited for the inverse operation taking a signal to its generating delta function – a deconvolution task. In previous studies, networks were shown to successfully embed traditional inverse iterative algorithms into layered CNNs [57, 58]. In addition, CNNs can be effective for modeling noise inflicted by detector sensitivity and exhaustion, as demonstrated by signal denoising networks [53, 54, 55]. Recently Li et al. [59] utilized a deep learning framework to separate overlapping ultrasound echoes and estimate the thickness of buried pipelines. Their solution is shown to improve TOFD estimation, however it suffers from several limitations. In their work, two networks are trained independently, solving several separate sub-problems. The first network segments the signal, including two overlapping echoes, into two separate signals. The second network is applied to each signal in isolation and finds the echo's TOF. Both networks used are rather heavy-weighted considering the needs of the problem, rendering slow training and inference.

In this paper, we suggest a simple network which solves the problem end-to-end as a detection problem. We construct a lean, yet effective, deep network for the overlapping echoes detection task. Unlike the method of [59], which is limited to two overlapping echoes, our proposed network solves a general detection problem, with unknown number of overlapping echoes and varying echo patterns. The proposed CNN framework is composed of long filters in the first layer to simulate the deconvolution of the signal back to its origin delta, followed by several layers with short filters to project the deconvolved activations to the space of clean and sparse sequences with few active delta functions. The training process is conducted with a simulated dataset, created based on a physical model of the ultrasound echoes, with significant variance in echo parameters, noise and overlaps.



The proposed network is compared to competing methods on simulated test datasets and on two physical phantoms taken from different materials. The network achieves high detection rates in adverse conditions including high echo pattern variance and severe successive overlaps. Our simulation experiments show that the network is preferable to competing algorithms in a wide range of signal overlap and SNR levels, and enables more accurate detection. With the physical phantoms, the network was tested and compared to competing algorithms at the task of layer thickness estimation from ultrasonic scans. The results show the network enables higher axial resolution, and provides more stable and accurate prediction than competing methods.

## 3.2 Problem Formulation and Related Work

Traditional signal processing methods model the measured signal  $y(t)$  using a linear time-invariant convolutional model of the form

$$y(t) = d(t) * x(t) + e(t), \quad (3.1)$$

where  $*$  denotes the linear convolution operator,  $d(t)$  is the transmitted ultrasonic pulse,  $x(t)$  is the inspected object reflectivity, and  $e(t)$  is a Gaussian noise function. Deconvolution of  $x(t)$  given  $y(t)$  and  $d(t)$  is an ill-conditioned problem since  $d(t)$  is usually a narrow-band signal. Therefore, one should enforce some prior on  $x(t)$  to obtain de-convolution results with a physical meaning. For the ultrasound application, strong echoes are expected only from interface locations between two materials with different impedance. Hence  $x(t)$  is comprised mainly of zeros, and a sparsity constraint on  $x(t)$  makes a good prior. The common way to enforce this constraint is by using the "sparse spike train" model [60, 61, 47]

$$y(t) = d(t) * \left\{ \sum_{m=1}^M x_m \delta(t - \tau_m) \right\} + e(t), \quad (3.2)$$

where  $\delta(t)$  is the Dirac delta-function,  $M$  is the number of echoes, and  $x_m, \tau_m$  are the amplitude and time of flight of the  $m^{\text{th}}$  echo. Model (3.2) defines all the echoes to be of the same form  $x_m d(t - \tau_m)$ , hence it can only be used when the attenuation is negligible and pulse shape does not significantly vary with time. A more flexible model [43], where each individual echo may have a unique shape, can be defined by

$$y(t) = \sum_{m=1}^M x_m d(\theta_m, t - \tau_m) + e(t). \quad (3.3)$$

Here,  $d(\boldsymbol{\theta}_m, t)$  represents the  $m^{\text{th}}$  ultrasonic pulse shape, modeled by the parameters  $\boldsymbol{\theta}_m$ , and pulse shape is allowed to vary among echos.

Traditionally,  $\{x_m, \tau_m, \boldsymbol{\theta}_m\}_{m=1}^M$  is recovered from  $y(t)$  using dictionary-based methods. This is achieved by seeking a sparse vector  $\mathbf{x}$  satisfying the relationship  $\mathbf{y} = \mathbf{D}\mathbf{x} + \mathbf{e}$  where  $\mathbf{D} \in C^{n \times m}$  is an overcomplete dictionary composed of  $\{\Phi_i\}_{i=1}^m$  atoms in columns, with  $\Phi_i \in C^n$  and  $m > n$ . Matching Pursuit (MP) [45] greedily constructs successive approximations of the signal by finding maximal projections of the signal residual on atoms of  $\mathbf{D}$ . Orthogonal Matching Pursuit (OMP) [46] improves MP by adding a least-square minimization to each step of MP. Support Matching Pursuit (SMP) [47], specifically designed for resolving ultrasonics overlapping echoes, extends MP by adding in the iterative stage a relaxed support measure corresponding to the p-norm with  $0 < p < 1$ . Another method for solving sparse problems is the Iterative Shrinkage Thresholding Algorithm (ISTA) [48]. This algorithm iterates between two operations: linear estimation, and shrinkage based on a soft thresholding function.

### 3.3 Echo Detection using Deep Networks

A network is trained for echo detection using a dataset  $\{(\mathbf{y}_i, \mathbf{x}_i)\}_{i=1}^N$  of signal examples  $\mathbf{y}_i \in R^{T_s}$  along with their respective ground truth delta-train reconstructions  $\mathbf{x}_i \in R^{T_s}$ . The network is a parametric inverse model  $h_{\mathbf{W}}(\mathbf{y}) : R^{T_s} \rightarrow R^{T_s}$ , accepting a signal example  $\mathbf{y}$  as input and parametrized by a set of weight parameters  $\mathbf{W}$ . Training consists in minimizing a loss function over the choice of  $\mathbf{W}$

$$\mathbf{W}^* = \underset{\mathbf{W}}{\operatorname{argmin}} \sum_{i=1}^N L(h_{\mathbf{W}}(\mathbf{y}_i), \mathbf{x}_i) + J(\mathbf{W}). \quad (3.4)$$

Here  $L$  is a loss function between the ground truth and the model prediction, and  $J$  is a regularization term posing limitations on the model parameters to reduce overfit. In the following sections we describe the details of the dataset, the model  $h_{\mathbf{W}}(y)$ , and the training process.

#### 3.3.1 Data Simulation

The physical model used for generation of a single echo is described in 3.3.1.1, followed by description of signals and outputs in 3.3.1.2.

##### 3.3.1.1 Gaussian Echo Model

Various parametric models have been proposed to model the echo pattern  $d(\boldsymbol{\theta}_m, t)$  described in Eq. (3.3) [43, 40, 62, 63]. We use a model describing the ultrasonic echo with parameters  $\boldsymbol{\theta}_m = [\omega, \phi, \sigma]$ , associated with physical parameters of the pulse as follows

$$d(\boldsymbol{\theta}, t - \tau) = K_{\theta} g(t - \tau, \sigma) \cos[\omega(t - \tau) + \phi], \quad (3.5)$$

which is the product of a Gaussian envelop function with a harmonic function.  $g(t - \tau, \sigma)$  is a Gaussian envelop, with  $\sigma$  defining the scale (proportional to the pulse-width) of the echo around its arrival time  $\tau$ .  $\cos[\omega(t - \tau) + \phi]$  defines an harmonic function with  $\phi$  and  $\omega$  its phase and center-frequency. The constant  $K_{\theta}$  ensures that  $\|d(\boldsymbol{\theta}, t - \tau)\|_2 = 1$ .

### 3.3.1.2 Dataset Randomization

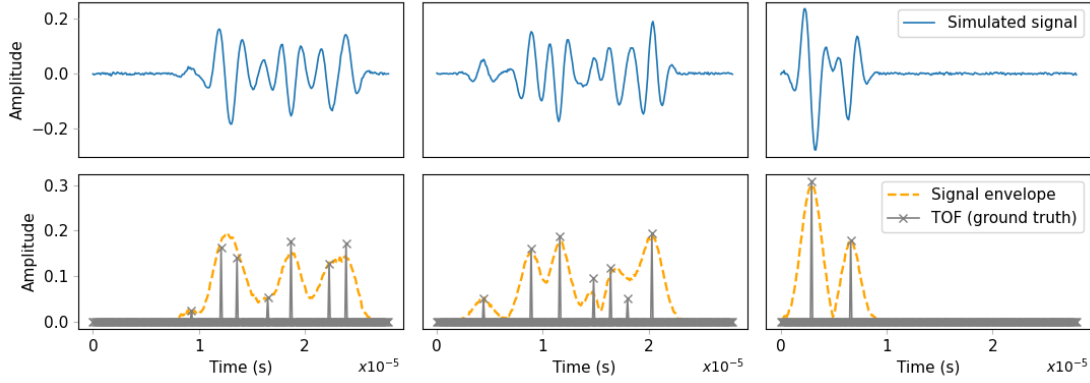
Signal examples  $\mathbf{y}$  are created as superpositions (Eq. 3.3) of Gaussian echoes (Eq. 3.5). For each generated signal  $\mathbf{y}$ , a target sparse train of delta functions  $x(t) = \sum_{m=1}^M x_m \delta(t - \tau_m)$  is generated as the reconstruction target. Examples of the simulated signals used for network training and testing are demonstrated in Figure 3.2.

**Echo pattern randomization.** Echo pattern parameters described in Eq. 3.5,  $\boldsymbol{\theta} = [\omega, \phi, \sigma]$ , are drawn from uniform distributions. The phase  $\phi$  is drawn in  $[0, 2\pi]$ . The center frequency  $\omega$  and the signal width  $\sigma$  are drawn in  $[\omega_0, \omega_r], [\sigma_0, \sigma_r]$  respectively, where  $\omega_0, \sigma_0$  are the center frequency and pulse width measured from a transducer reference echo-signal received from a flat reflector immersed in water. The parameters  $\omega_r, \sigma_r$  control the width of the echo distributions. Specific values and distributions are described in the Section 3.4.

**Full signal randomization.** The number of echoes in a signal is drawn from a uniform distribution in the range  $[1, K]$ . In order to create significant overlap between echoes in the dataset, the TOFD between neighbouring echoes is drawn from an exponential distribution. The first echo TOF is randomly drawn uniformly from the  $T_s$  time-samples, and the following echoes' TOFD is drawn from  $\exp(\lambda)$ .  $\lambda$  hence defines the task overlap difficulty, and can be adapted to the task needs. In our experiments, we set  $\lambda = \frac{1}{3\sigma_0}$ , and allow TOFD range of  $[1.5\sigma_0, 6\sigma_0]$ , with  $\sigma_0 = 9$  time-samples. The echo amplitude  $x$  is drawn uniformly in  $[\alpha, 1]$ , with  $\alpha$  a difficulty parameter controlling the echo's dynamic range. Noise is modeled in the signal by additive Gaussian noise. The level of noise is defined by  $SNR = 10 \log_{10}(\frac{P_s}{P_n})$ , with  $P_s$  the power of the signal (normalized to 1 in all experiments) and  $P_n$  is the power of the noise. The desired noise level is set by tuning  $P_n$ .

## 3.3.2 Network Design

The network  $h_W(y) : R^{T_s} \rightarrow R^{T_s}$  is a Fully Convolutional Neural Network (FCN) [64]. It receives as input the signal  $y(t)$  of  $T_s$  time stamps, and outputs a prediction  $x(t)$  of the same length. The proposed network, illustrated in Fig. 3.3, consists of two main parts: an inverse operator and a projection operator. The inverse operator is a single wide convolution layer, performing correlations with multiple learned pulses of the signal with multiple pulse shapes. The projection is performed



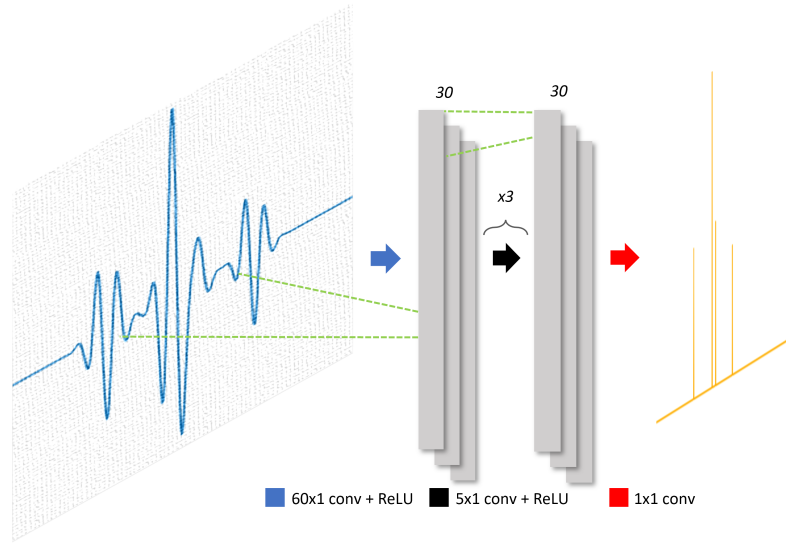
**Figure 3.2: Simulated data illustration.** Examples of Simulated signals (top) used for network training and testing, along with their envelopes and TOFs (bottom).

by several convolutional layers with small-sized filters, projecting the noisy result to the space of clean, sparse and separated deltas.

**Inverse operator.** Since the signal generation presented in Eq. (3.3) defines a superposition of echos with variance in their shape, its inverse function can be modeled by a standard convolution layer with multiple filters whose size is large enough to cover the echo’s length. The wide convolution layer is employed with a filter size of  $6\sigma_0$ , the approximated length of the generating pulses, and depth  $f$ , the number of representative filters, set in the following experiments to 30. The input is padded with  $3\sigma_0$  for the output to obtain the length of the input signal  $T_s$ . The wide convolution is followed by a ReLU [65] activation.

To observe the effect of first layer filters, the network was trained separately with different number of filters and representative filters were inspected. In Figure 3.4, representative first layer filters are presented. In Figure 3.4 (left), a filter from a network trained with a single filter is shown. With one filter, the network functions as an adaptive non-linear filter (with ReLU layers providing non-linearity). The filter learns a sinusoidal shape with a minor peak at its center, which is suitable to locate an echo. However, this alone is not sufficient to resolve overlapping echoes of different patterns, and indeed a network with a single filter produces poor overlap separation results (over a 55% decrease in echo detection accuracy compared to network results described in Section 3.5.1). In Figure 3.4 (right), filters with the largest norms from a network with 30 filters in the first layer filters are presented. One can observe that the learned filters reflect the temporal oscillatory nature of the received signals related to the signal bandwidth and central frequency. Multiple filters adaptively and simultaneously learn to identify different time-variant oscillatory shapes, information that is later integrated by the network for the overlap separation prediction.

**Projection operator.** After deconvolution, multiple maps may exhibit strong response to the same echo reflection, as a single echo is detected by multiple filters. However, in the desired output only a single exact TOF of a reflection is represented by a non zero value. To enable reasoning, competition among neighbouring components and removal of redundant noise, the inverse operator



**Figure. 3.3:** Schematic overview of the suggested US-CNN architecture.

is followed by a series of 3 conv layers. These layers use a kernel of size 5, stride 1 and dilation [50, 55] of 2, and each is followed by a ReLU activation. The last convolution layer is followed by a dropout layer [66], employed to improve network robustness to overfitting, with the dropout rate set to 0.2. Finally, the activations are transferred into  $1 \times 1$  convolution to reduce the multiple maps representing each time-sample into a single output.

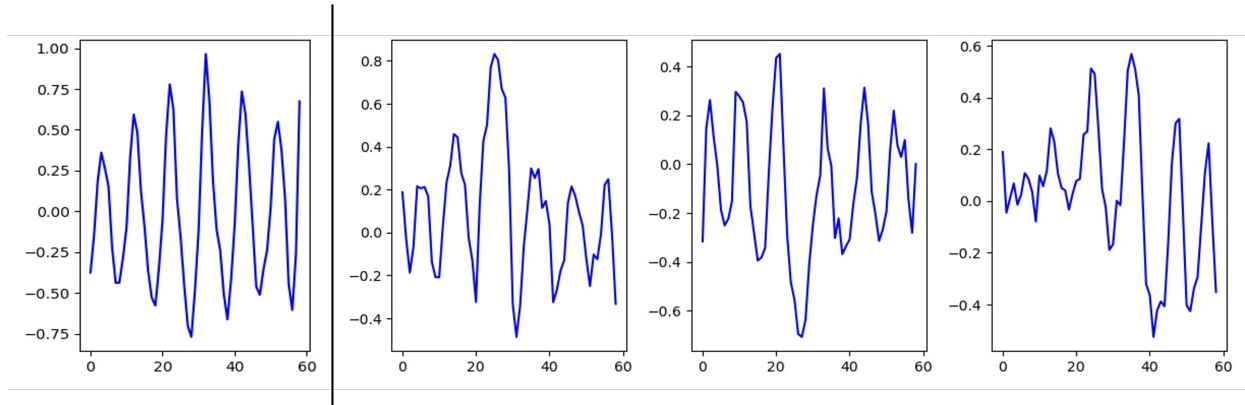
### 3.3.3 Network Training

We pose echo detection as a dense regression task – for each time sample, the model predicts the amplitude of the echo reflection at that point, if there was any. Given  $N$  training signals  $\{y_1, \dots, y_N\}$ , the network is trained using a dense  $l_2$  loss

$$\mathcal{L}_{MSE} = \sum_{i=1}^N \sum_{t=1}^{T_s} (h_W(y_i(t)) - \hat{x}_i(t))^2. \quad (3.6)$$

where  $\hat{x}_i$  are soft versions of  $x_i$  described below.

**Gaussians for prediction relaxation.** The ground truth  $x(t)$  is a sparse vector, with few non zero values representing the echoes' TOFs. Estimation of such non-continuous target vectors is a difficult task to learn. Deviation of only one time-sample in prediction cause an error much bigger than if the prediction was zero, even though the prediction is almost precise. This discourages learning of useful, non trivially-zero predictions. To enable a smoother energy landscape for the learning problem, we propose to smooth  $x(t)$  by convolving it with a zero mean Gaussian kernel  $\hat{x}(t) = g(0, \alpha\sigma) * x(t)$ , where  $g(0, \alpha\sigma)$  is a Gaussian filter of variance  $\alpha\sigma$ .  $\sigma$  represents the assumed signal echo-width, and  $\alpha$  is a task dependant hyper-parameter defining the width of the smoothed



**Figure. 3.4:** Representative filters learned by the first layer of the network. **Left:** Filter of a network trained with one filter in the first layer. **Right:** 3 Representative filters of a network trained with 30 filters in the first layer.

ground truth, set in our experiments to 0.2. This idea is similar to the generation of "heatmaps" as regression targets in computer vision tasks [67, 68]. Exact echo locations can still be extracted from the network prediction by locating the highest prediction peaks.

## 3.4 Experimental Setup

Experiments were carried on simulated data and two real phantoms. Detection accuracy was first measured on simulated data, where data can be generated flexibly with exact ground truth. The method was then applied to the layer thickness estimation task in two real-world ultrasonic scans of physical phantoms, specifically designed for tests of this nature.

**Data Simulation.** The training set consisted of 10,000 simulated signals, divided 80%-20% between the train and validation sets. Each signal contains 500 time-samples. Three types of training sets were generated. The first set is drawn without parameter randomization of  $\sigma, \omega$  according to the randomization procedure proposed in Section 3.3.1.2. The second set was drawn with narrow distribution of  $\sigma, \omega$  and the third with a wide distribution. The pulse width was randomized from  $\sigma_0$  to  $1.2\sigma_0$  for the narrow distribution and  $1.4\sigma_0$  for the wide distribution. The center frequency was randomized according to the -3dB bandwidth of the transducer for the narrow distribution and -6dB bandwidth for the wide distribution. Table 3.1 shows the parameters of the three scenarios. In all experiments, the maximal number of echoes  $K$  was set to 8 and the minimal echo amplitude parameter  $\alpha$  was 0.1.

**Network Training.** Training was done with the ADAM gradient-based optimization method [69] for 80 epochs with a learning rate of 0.01 and batch size of 150. Learning rate was reduced by a factor of 2 if the validation error did not improve for 12 epochs and training proceeded until validation error did not improve for 20 epochs. The network was implemented in Pytorch [70], and trained for about 2 minutes on an NVidia GeForce TitanX GPU.

**Table 3.1:** Echo pattern parameter distributions in the simulated datasets.

Parameter\Distribution	None	Narrow	Wide
Echo width $\sigma$ (time-samples)	$\sigma_0 = 9$	[9, 10.8]	[9, 13]
Center frequency $\omega$ (MHz)	$\omega_0 = 5.4$	[4, 5.4]	[3, 5.4]

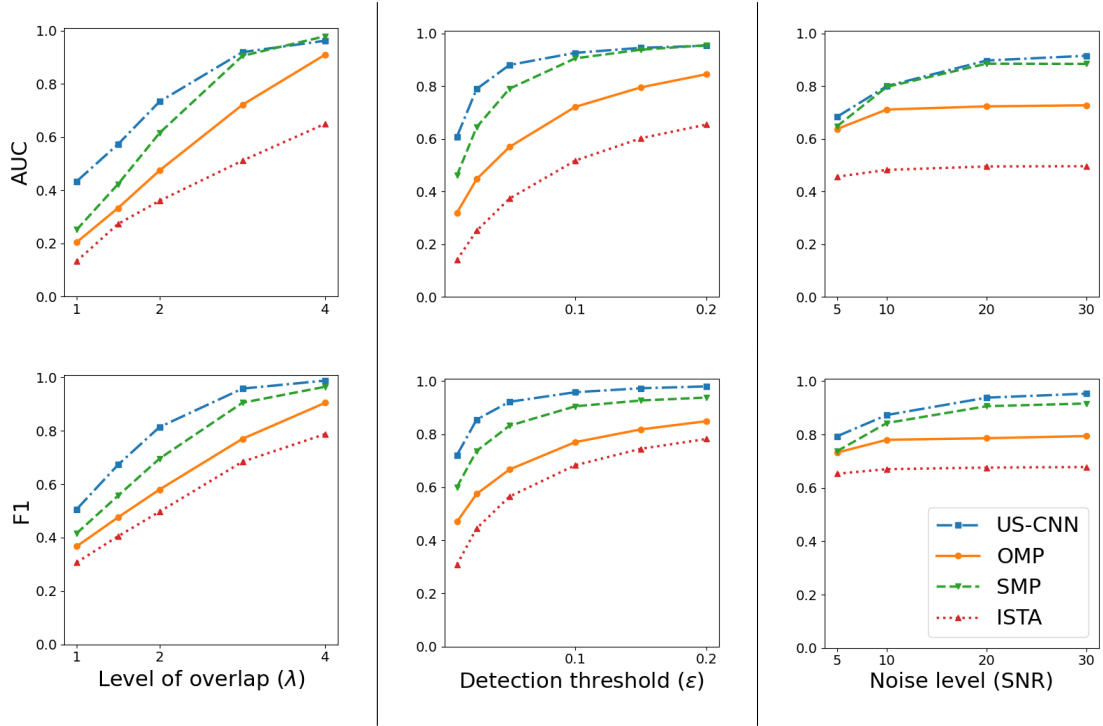
**Competition.** In all experiments, the network is compared to standard SSR methods used for solving the echo detection task – OMP [46], SMP [47] and ISTA [48]. To handle variations in the pulse phase, dictionary atoms were constructed based on a complex variant of Eq. 3.5 [47]. Dictionary parameters were determined differently for each of the 3 data distributions described in Table 3.1. The number of pulse width and center frequency values represented in the dictionary were set to 1, 3, and 6 for the no randomization, randomization with narrow distribution, and randomization with wide distribution respectively, as more values are required to cover the wider echo distributions. For all distributions, the number of TOF values represented in the dictionary are equal to the number of sampling times  $T_s = 500$ . The dictionary included the Cartesian product of pulse width, center frequency and TOF values, so it contains 500, 4500 and 18000 values for None, Narrow and Wide distributions.

Stopping conditions were optimized separately for simulation and real phantom experiments, to provide each method with the best obtainable results. In simulation, where there are up to 8 echoes in a signal, OMP and SMP ran until the residue energy is below 3% of the measured signal energy with a hard limit of 15 iterations. On the real phantoms, OMP and SMP methods ran until the residue energy is below 25% of original energy with a maximum of 4 iterations. This early stopping criterion is used in order to reconstruct only significant echoes. ISTA runs for 3 iterations with a threshold coefficient of 0.1.

**Post Processing.** Simple post-processing is introduced to reduce noise and promote sparsity in predictions. First, activations with values lower than a pre-defined value  $\beta$  are set to zero. For the physical phantoms, we set the threshold  $\beta$  rather high in order to catch the first two echoes. If there are less than two values above  $\beta$ , it is iteratively reduced by assigning  $\beta = c \cdot \beta$  with  $c \in [0, 1]$ , until two echoes are detected. In simulation,  $\beta$  is set to 0.01, while with the real phantoms it is set to 0.2 and  $c = 0.8$ . After the thresholding operation, non-maximum suppression (NMS) is applied to remove small detection activations which are near larger activations. The NMS range used is 14 time samples from the selected detection, which is approximately equal to  $1.5\sigma_0$ .

## 3.5 Results





**Figure. 3.5: Simulation results.** Comparison of detection accuracy as measured by AUC (top) and F1 (bottom) among the four methods tested. Detection accuracy is plotted as a function of echo overlap parameter  $\lambda$  (right), required accuracy threshold  $\epsilon$  (middle) and SNR (left).

### 3.5.1 Simulation Results

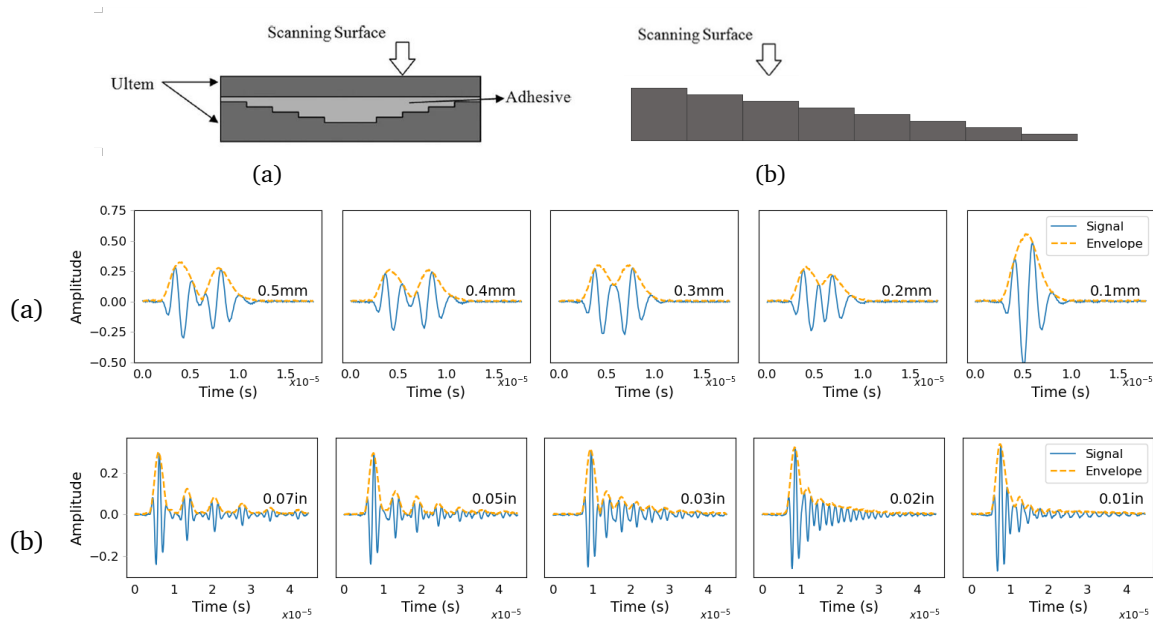
**Evaluation metric.** To evaluate accuracy in simulation, each detection (an output activation larger than zero) is compared with all ground truth delta-functions of the signal, to see if a hit can be found. Denote a detection hypothesis and a ground truth delta by  $z(t) = a_z \delta(t - t_z)$  and  $x(t) = a_x \delta(t - t_x)$  respectively, where  $a_z, a_x$  are the amplitudes and  $t_z, t_x$  are the delta function timings. A detection is declared a 'hit' if

$$\frac{\sum_{t=1}^T (s(z(t)) - s(x(t)))^2}{\sum_{t=1}^T (s(x(t)))^2} < \epsilon, \quad (3.7)$$

where  $s(\cdot)$  returns the Gaussian envelope component of the simulated signal, as described by Eq. (3.5). Based on this hit definition, a recall-precision curve over the test set is plotted and the area under the curve (AUC) and maximal F1 score (across all threshold values) are reported as the accuracy indices. Note that a detection is classified as a hit only if it matches a true echo in both location and amplitude. For small values of  $\epsilon$ , this can be rather challenging.

**Results.** We report accuracy on the narrowly distributed (scenario 2 in Table 3.1) test set as a function of three main difficulty parameters:  $\lambda$  (Fig. 3.5 (left)) controlling echo overlap, required detection accuracy  $\epsilon$  (Fig. 3.5 (middle)), and SNR (Fig. 3.5 (right)), defined in Section 3.3.1.2. The default values of  $\lambda, \epsilon$  and SNR in these experiments are set to 3, 0.1, and 20 respectively, and in each experiment a single parameter changes while the other two are kept fixed. For each algorithmic





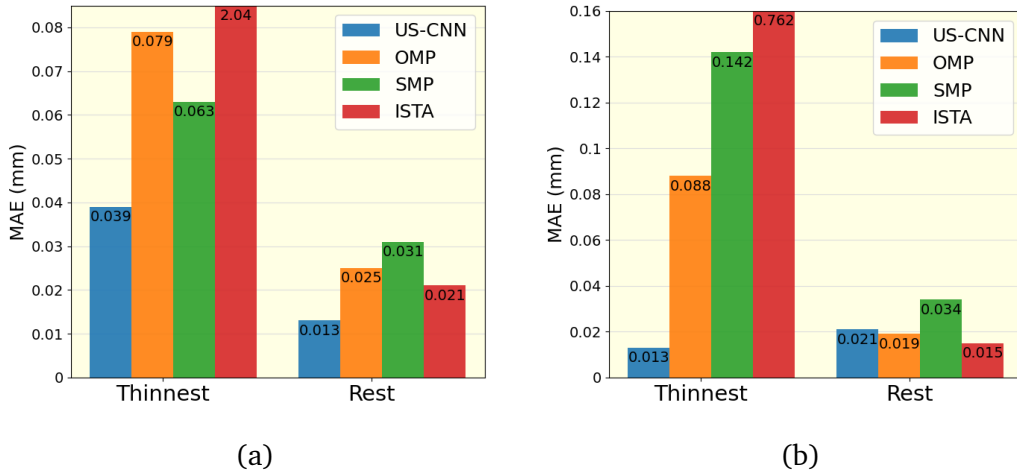
**Figure. 3.6: Demonstration of real phantoms used for thickness estimation experimentation. Top:** Side view illustrations of the phantoms: (a) Round Ultem phantom, and (b) Aluminum phantom. **Bottom:** Time-domain signals from different layers of the (a) Ultem and (b) Aluminum phantoms.

method, variants corresponding to the 3 distributions (see Table 3.1) were tested and results are shown for the best option (narrow distribution for US-CNN, non-randomized dictionaries for the other methods).

As can be seen in Fig. 3.5, the network provides higher accuracy than competing algorithms in most of the scenarios. The left graph shows that the network has a considerable advantage over competition for scenarios with significant echo overlap ( $\lambda$  values of 1 and 2). For high overlap of  $\lambda = 1$  it provides almost twice the accuracy than alternative algorithms. The middle graph shows that the network has a significant advantage when high detection accuracy is required according to Eq. (3.7) (small  $\epsilon$  values). SMP, a greedy search algorithm tailored specifically for the task of echo separation, provides similar results to the network's in easier cases. When accuracy as a function of SNR is considered (Fig. 3.5 (right)), the network and SMP are the leading algorithms.

### 3.5.2 Real Phantom Results

We assess the network on scans taken from two physical phantoms in the task of layer thickness estimation. The first phantom is composed of two disks made from Ultem (a type of machinable polymer) attached by an adhesive layer (see Fig. 3.6(a)). The top Ultem disk is flat and the other has inner circular steps of 0.1mm in height. The two disks are attached with adhesive, forming an adhesive layer of thickness ranging from 0.1mm to 0.5mm at intervals of 0.1 mm, according to the steps in the lower Ultem disk. The second phantom is a thin 8-step Aluminum calibration block, with thickness ranging from 0.03 inch to 0.1 inch at intervals of 0.01 inch (see Fig. 3.6(b)). Both



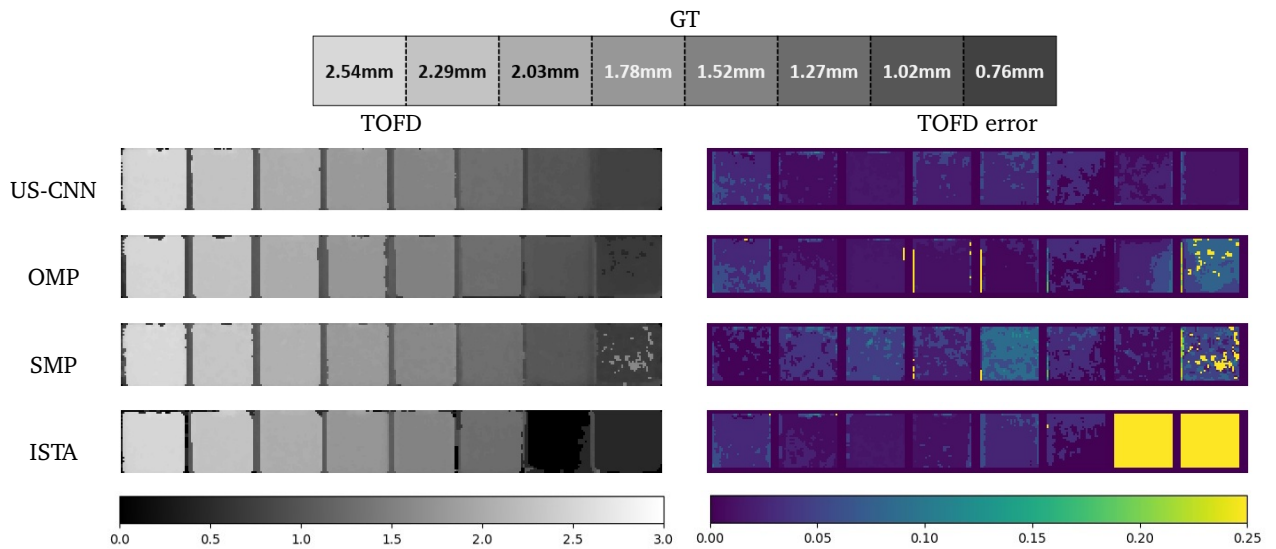
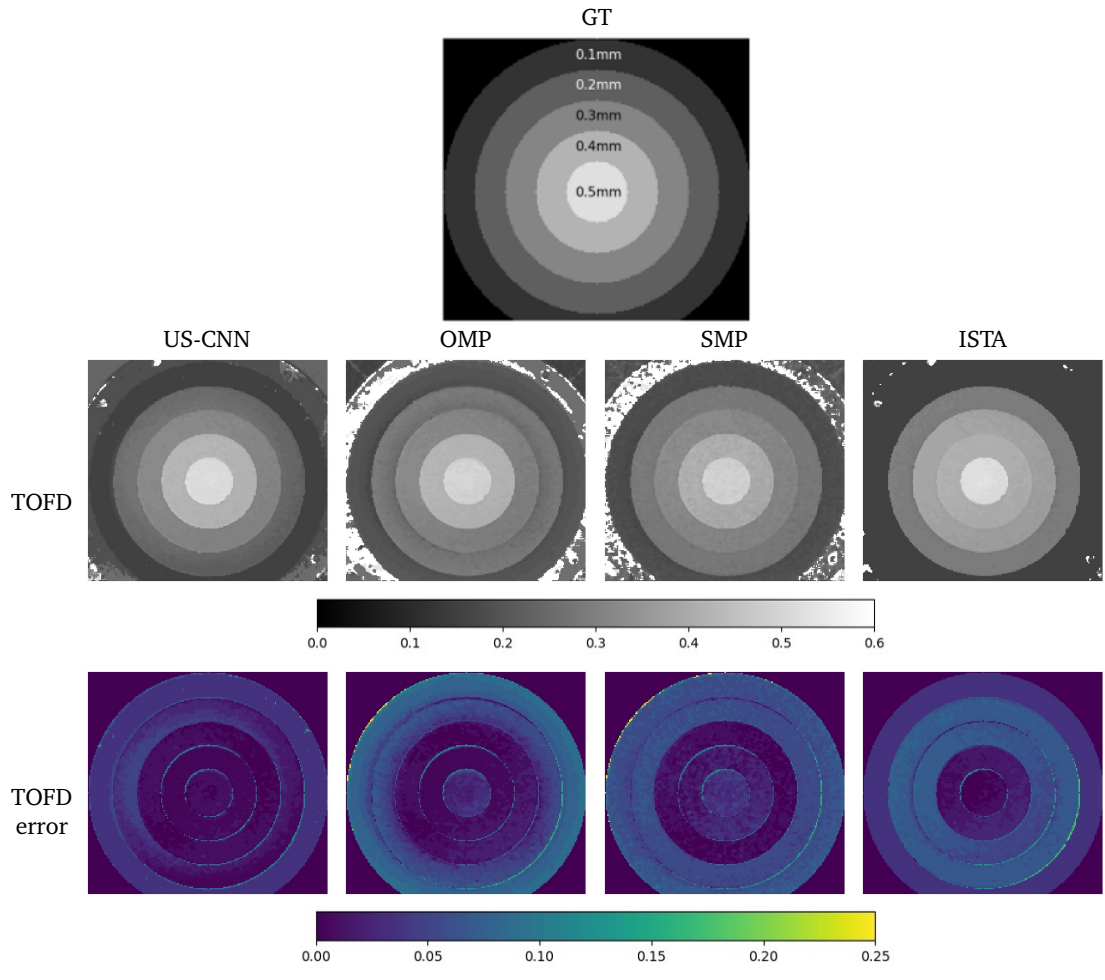
**Figure 3.7: Thickness estimation results.** Mean Absolute Error (MAE, in mm) of the tested methods on the (a) Ultem and (b) Aluminum phantoms. In each figure, the left histogram shows average error on the thinnest layer, the right histogram shows the average error of the remaining layers.

phantoms were scanned by a scanning acoustic microscope equipped with a 5MHz, flat, 0.5 inch diameter transducer.

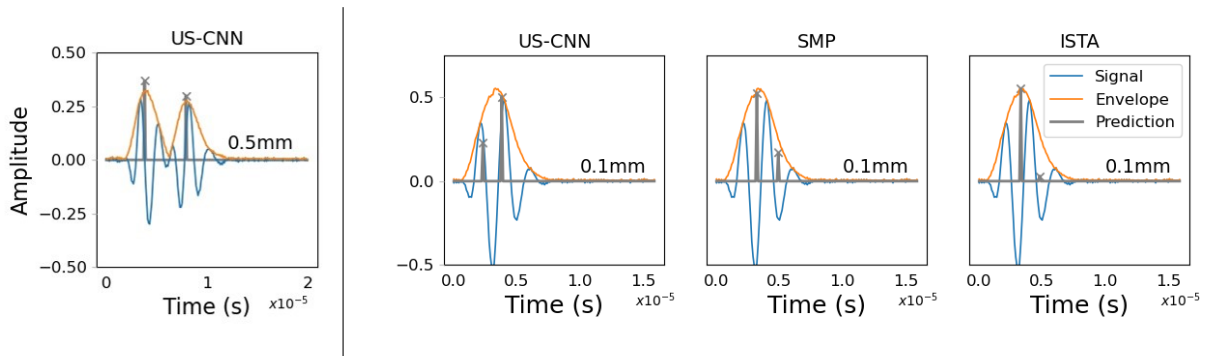
The two phantoms differ in material and structure, posing a different echo separation challenge. First, the change in material and experimentation time leads to differences in the reflected echoes in terms of the echo patterns. Additionally, the aluminum block introduces reverberating multi reflections that result in additional overlaps after the second echo.

**Thickness estimation results.** Layer thickness estimations were obtained by extracting the TOFD between the two largest echoes and converting it from time (time-samples in microseconds) to distance (mm) based on the given material sound-velocity of the two materials. The inspected phantoms were premeditatedly designed and manufactured with a high level of precision for experimentation. Therefore, the exact distance in each thickness is reliably known. Fig. 3.7 shows the mean absolute error (MAE) obtained by all methods for the thinnest layer, and for the average of the remaining layers in the two phantoms. The thinnest layer provides information regarding the range resolution of each method, whereas the average of the remaining layers measures the overall detection accuracy and consistency. For each algorithmic method, variants corresponding to the 3 distributions (see Table 3.1) were tested and results are shown for the best option (narrow distribution for US-CNN, non-randomized dictionaries for OMP and SMP, and wide distribution for ISTA).

For the thinnest layers in each model, where the neighboring echoes severely overlap, US-CNN provides the best accuracy on both models, and with a significant margin. For the Ultem phantom, the network also shows the lowest error rate overall across the different layers, whereas for the Aluminum phantom ISTA provides the lower error rates. However, for the Aluminum phantom all methods successfully separated the overlapping echoes across all layers but the thinnest. This



**Figure 3.8: Thickness estimation visualization for layer physical phantoms.** Each pixel within the Time-of-Flight-Difference (TOFD) predictions (Gray scale images) indicates the estimated thickness for the spatial location. The color images are the corresponding TOFD error between the prediction and the ground truth. **Top:** Ultem phantom. **Bottom:** Aluminum phantom.

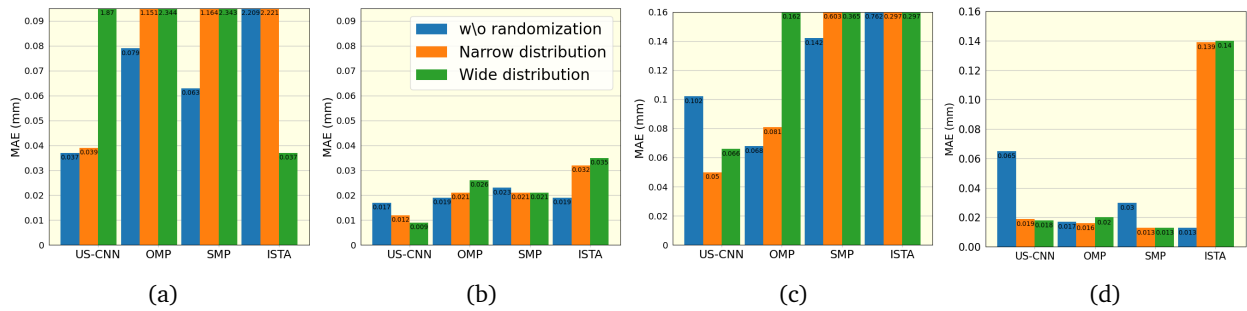


**Figure. 3.9: A-scan prediction visualization.** **Left:** A-scan with US-CNN echo detection from the 0.5mm ring of the round Ultem phantom. **Right:** A-scan from the 0.1mm ring of the round Ultem along with detections of US-CNN, SMP and ISTA.

can be seen by the low MAE values, ranging from 0.015mm to 0.034mm, comparing to the block thicknesses, ranging from 1mm to 2.54mm, representing a maximal deviation of 3% from the correct value.

**Thickness estimation visualization.** Figure 3.8 shows the detection results of the tested algorithms as two dimensional thickness maps. While all the methods perform reasonably well qualitatively for thick layers, the network provides a more accurate and stable prediction. For the thinnest layers, the network shows significant improvement over competing algorithms, as can be clearly seen by the TOFD error figures. US-CNN successfully separates overlapping echoes in the thinnest layer of both phantoms, enabling a new range resolution for layer thickness estimation. The fine differences between the methods for thin layers can be better seen in a closer inspection of the A-scans. In Fig. 3.9, A-scans and predictions of the tested algorithms on the round model are shown. On the left an a-scan from the 0.5mm ring of the phantom is shown, along with US-CNN prediction and the signal envelope. As can be seen, the prediction is located at the center of the signal envelope. On the right, prediction of an a-scan from the 0.1mm ring is shown for US-CNN, SMP, and ISTA. As can be seen, SMP and the ISTA methods misplace a single large echo in the center the overlapping echoes' combination. This main central middle detection is followed by additional one or more detections corresponding to small residual tails. These methods hence fail to detect the accurate interface positions.

**Amenability to signal variation.** In Figure 3.10, layer thickness MAE of the tested method is shown as a function of training distribution width. Results are shown separately for overall estimation (where the mean is over all estimations in all layer thicknesses) and for the thinnest layer in each model. For SSR methods, usually wider dictionaries used in wider distributions effectively causes a deterioration in thickness estimation. The network exhibits another behavior. For overall performance, the network seems to monotonically gain from training on a wider distribution. For the thinnest layers with the highest overlap, the widest distribution causes damage, but a distribution with moderate variance provides the best accuracy. These results suggest that the network is a more plausible alternative when large echo variance is present.



**Figure 3.10: Error as a function of training distribution width.** Mean absolute errors of the four tested methods as a function of training distribution width. (a)-(b) **Round phantom.** MAE of (a) thinnest layer and (b) average of the rest of the layers. (c)-(d) **Rectangular phantom.** MAE of (c) thinnest layer and (d) average of the rest of the layers. Very large errors (full height bars) are measured when a method fails to find a second echo relevant for width estimation.

**Inference efficiency.** Network inference for a single simulated signal, containing 500 time samples, takes 0.5 millisecond on GPU and 1.5 milliseconds on CPU. Processing the signal in an unoptimized manner with ISTA, OMP, and SMP, used with the minimal number of dictionary atoms, takes 5, 10 and 20 milliseconds respectively on CPU. That is, US-CNN is 3.5-40 times faster than traditional algorithms.

## 3.6 Conclusions

In this paper a lean and efficient neural network is presented for the task of overlapping echo detection. sparse approximation techniques in the task of overlapping echo detection. Using simulated data, the network was shown to provide better accuracy than competition in high overlap conditions and when high detection accuracy is required. Tested on two physical phantoms, the network provided with accurate and stable estimation in a layer thickness estimation task. Experiments with increased echo variance in the training phase showed that compared to dictionary-based models, the network provides a more robust alternative for coping with high echo variance. In further work, the network model can be adapted to additional signal physical characteristics, such as chirps and multi-reflections, simply by adding those to the training data.

**Acknowledgements:** This work is supported by the Pazy foundation Israel, through grant no. ID63-2018.

# Anomaly Detection for High-Content Image-Based Phenotypic Cell Profiling

## In Preparation and Submission:

- **Shpigler, A.**, Kolet, N., Golan, S., Weisbart, E. & Zaritsky, A. Anomaly Detection for High-Content Image-Based Phenotypic Cell Profiling.

## Abstract:

High-content image-based phenotypic profiling combines automated microscopy and analysis to identify phenotypic alterations in cell morphology and provide insight into the cell's physiological state. Classical representations of the phenotypic profile can not capture the full underlying complexity in cell organization, while recent weakly machine-learning based representation-learning methods are hard to biologically interpret. We used the abundance of control wells to learn the in-distribution of control experiments and use it to formulate a self-supervised reconstruction anomaly-based representation that encodes the intricate morphological inter-feature dependencies among classical representations while preserving the representation interpretability. The performance of our anomaly-based representations was evaluated for downstream tasks with respect to two classical representations across four public Cell Painting datasets. Anomaly-based representations improved reproducibility, Mechanism of Action classification, and complemented classical representations. Unsupervised explainability of autoencoder-based anomalies identified specific inter-feature dependencies causing anomalies. The general concept of anomaly-based representations can be adapted to other applications in cell biology. This work is in the final stages of writing a manuscript that will be posted as a preprint on bioRxiv, and will be submitted to a top scientific journal in the field of computational biology.

## 4.1 Introduction

Visual cell phenotypes, characterized by the morphological features of a cell such as a cell shape and molecular composition, can serve as powerful readouts for cell state [71, 72, 73, 74, 75]. Alterations in visual cell phenotype, such as changes in cell shape and intracellular organization, can provide insight into the cell's physiological state, as well as assist in the diagnosis and treatment

of diseases [75, 76, 77, 78]. High-content image-based phenotypic profiling combines automated microscopy and automated image analysis to identify phenotypic alterations in cell morphology [79, 80, 81, 82, 83, 84]. For example, the Cell Painting assay uses high-content imaging, followed by analysis of multiple organelle stains in multiple cellular compartments at single cell resolution [85]. The most common approach for analysis of phenotypic profiling data, relies on pre-defined (a.k.a, “engineered”) single cell features, extracted with software tools such as Cell Profiler [86], and followed by aggregation of these single cell features across the cell population to represent a “phenotypic profile”. The deviation of the treatment-induced phenotypic profile is measured in relation to the profile of untreated/control cells. The phenotypic alteration defines the degree and direction of change in the cellular high-dimensional phenotypic space, and can be used for various applications such as discovering drugs that “shift” a “disease-associated” to a “healthy-associated” phenotype or finding chemical compounds with a phenotype associated with a desired mechanism of action (MoA) [87, 88, 83, 89].

The gold standard for measuring a phenotype relies on the (inaccurate) implicit assumption of independence between the features extracted from the imaged cells. One example for this approach is measuring the fraction of features that dramatically deviate from the control profile in response to a treatment [83, 90]. Another example is measuring the correlations between profiles of treatments’ replicates [83, 91, 92]. Of course, the profile features are interdependent, even after feature selection, because cell organization is so complex. For example, variation in cells’ intracellular organization may be largely explained by the cells’ shape [93]. Explicitly measuring these complex dependencies for all features is not feasible due to the curse of dimensionality [94]. Thus, the biological function investigated may be incorrectly interpreted due to a simplified representation of the underlying data complexity. Recent representation-learning methods train machine learning models to encode lower-dimensional cell-level or well-level embeddings (called “latent representations”) through weakly supervised learning [95, 96, 97, 98, 99, 100] or self supervised learning [101]. While representation-learning methods show promising results in terms of capturing the differences between treatments, current representation methods are not optimized to model the change posed by the treatment in respect to the control, but to distinguish between the different treatments (see Discussion). Moreover, representation-learning methods are harder to interpret, because the features forming the latent representations do not have a semantic cell biological explanation. Here, we propose using methods for anomaly detection to enhance the phenotypic profile representation in the context of high-content image-based phenotypic profiling by encoding intricate inter-feature dependencies while preserving the representation interpretability.

Anomaly detection aims at detecting abnormal observations that deviate from a predefined baseline pattern [102] and has vast applications in bioinformatics [103, 104], healthcare [105], and cybersecurity [106]. Anomaly detection relies on statistically characterizing the in-distribution of the data and defining observations that do not conform to this distribution as anomalous. Different approaches, such as neighbor-based [107, 108] and isolation-based [109] were applied for anomaly detection, with ML emerging as an especially powerful technique in recent years [110]. The advantage of ML methods, and particularly deep neural networks, stems from their ability to



integrate massive amounts of complex data into a generalized model that captures the inherent dependencies of the data distribution [111, 112]. High-content image-based phenotypic profiling naturally aligns with the formulation of an anomaly detection problem with many replicates of the control condition that can be used to statistically define the in-distribution baseline patterns, and a few replicates from each of the (many) treatments, that can be assessed according to their deviation from the baseline, or in other words how anomalous they are in respect to the in-distribution baseline patterns.

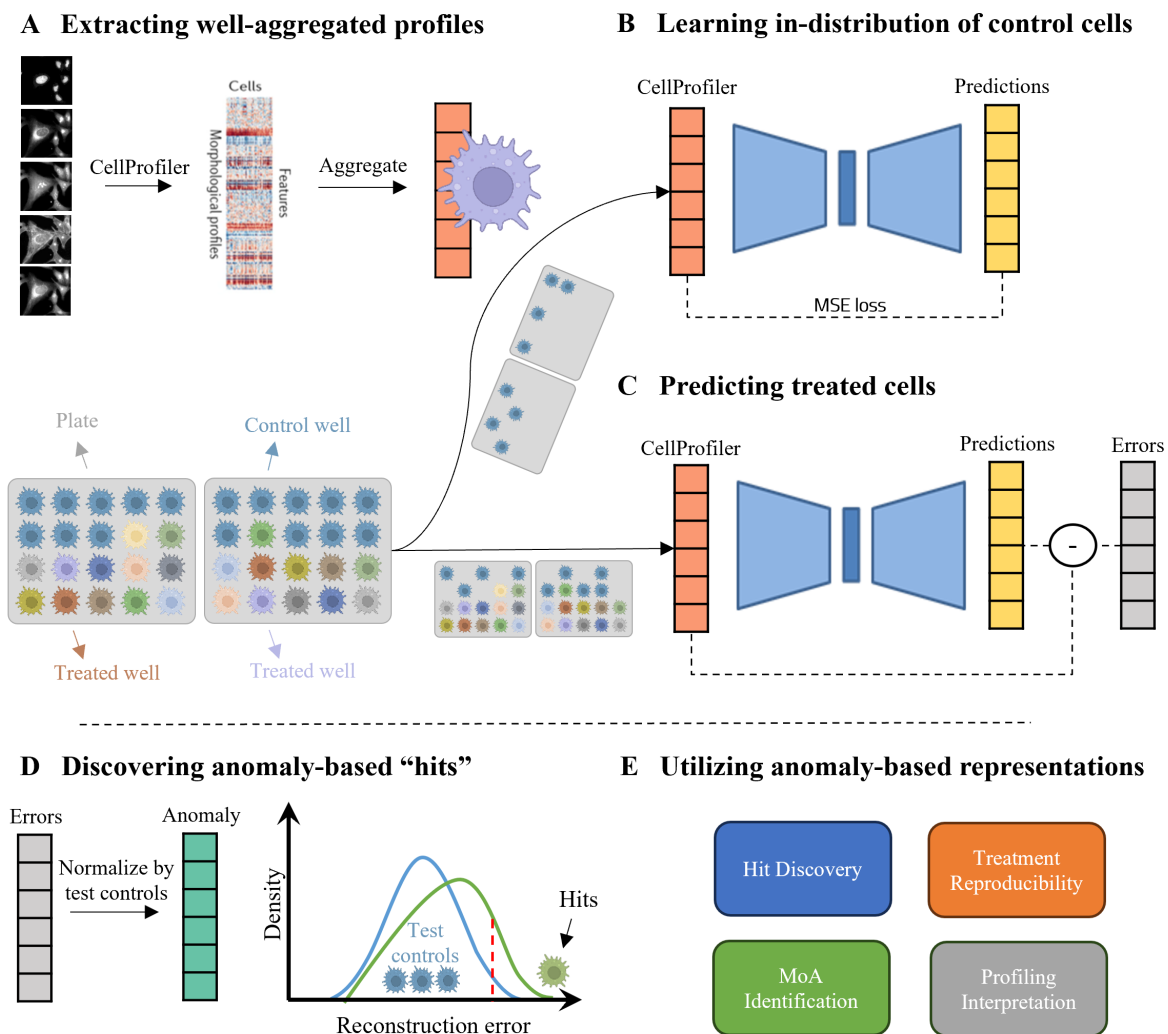
We present here a reconstruction-based, self-supervised, anomaly detection-based representation for high-content image-based phenotypic cell profiling using the “gold-standard” CellProfiler representation. Reconstruction-based methods for anomaly detection are trained to reconstruct “normal” (i.e., non-anomalous) samples from low-dimensional encodings under the intuition that anomalous samples will be less successfully reconstructed due to altered dependencies between the representation’s features. We demonstrated that our anomaly-based representations surpass the CellProfiler representations on multiple high-content Cell Painting datasets across different cell types and treatments [88], in terms of reproducibility and the downstream task of MoA identification. Moreover, we found that anomaly-based representations encapsulate complementary information in respect to the CellProfiler representations, leading to improved reproducibility and MoA identification. Finally, we demonstrated that applying an unsupervised method to explain autoencoder-based anomalies pinpointed specific inter-feature dependencies that changed to define an anomaly.

## 4.2 Results

### 4.2.1 Anomaly Detection Representation for Image-based Phenotypic Cell Profiling

Our anomaly detection-based method consists of three steps (Fig. 4.1). First, pre-processing, single cell feature extraction using CellProfiler[86] and well-level averaging across the cell population defining the well’s phenotypic profile (4.1A). Second, leveraging the abundance of control wells, to train an “in-distribution” autoencoder deep neural network that encodes a lower dimensionality compressed phenotypic profile and decodes it to reconstruct the input profile in the original dimensionality, according to the (in-distribution) baseline of the control wells’ profiles (Fig. 4.1B). Half of the control wells from each experimental plate were pooled for training, and used to standardize all wells from their respective plate. Following feature selection performed on all profiles, the train controls are used for training the “in-distribution” autoencoder to minimize the discrepancy between the input and the reconstructed profiles. The trained autoencoder learns non-linear interrelationships between the features in the control profiles. Having access to all plates and well locations, the in-distribution autoencoder was inherently guided to encode batch effects related to plate association, without the risk of exposing itself to data leakage, because the treated wells were not used for training. Third, the reconstruction errors of the in-distribution autoencoder





**Figure 4.1: Method: anomaly-based representations for cell profiling.** (A) **Top:** Well-aggregated profiles. Left-to-right: The CellProfiler software was used to extract single cell morphology features from Cell Painting images, the single cell features were aggregated to a well-level profile. **Bottom:** Each plate (gray) includes control (cyan) and treated (color) wells (cell icon). Half of the control wells were used to train the in-distribution autoencoder (B) and the other wells were used for evaluation (C). (B) The in-distribution autoencoder was trained to minimize the reconstruction error of control wells. (C) The in-distribution autoencoder was used to reconstruct control and treated wells. The reconstruction errors were calculated for each well. (D) The reconstruction error of a treated well was standardized according to the reconstruction errors of the control wells that were not used for training. These standardized reconstruction errors formed the “anomaly”-based representation. Treatments that lead to high reconstruction errors (green), in respect to the controls’ reconstruction errors (blue), were defined as “hits” (threshold in dashed red). (E) Anomaly-based representation can be used for a variety of downstream applications.

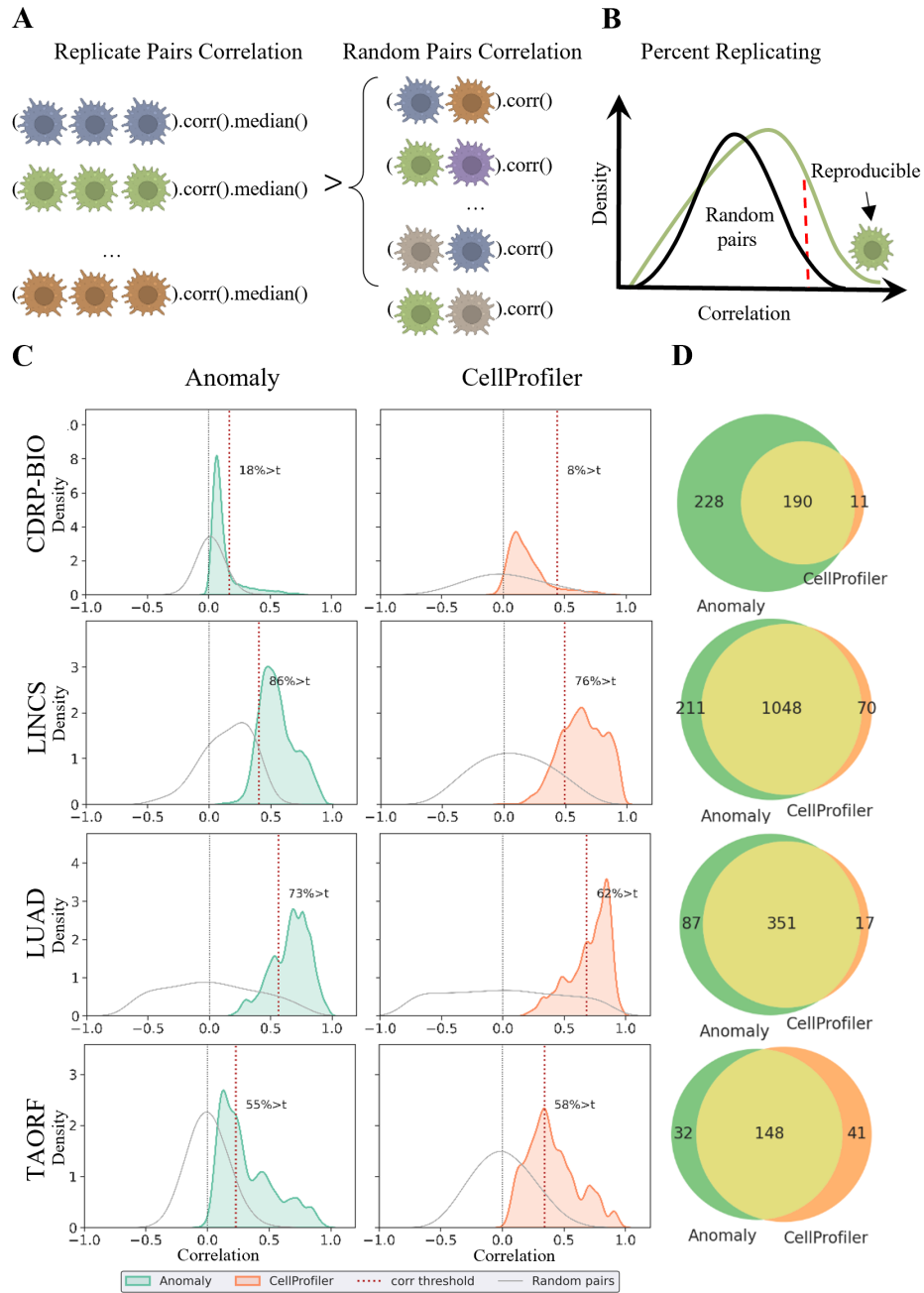
**Table 4.1: Datasets used in this study.** CDRP-bio (cpg0012-wawer-bioactivecompoundprofiling) [80], LINC (cpg0004-lincs) [83], LUAD (cpg0031-caicedo-cmvip) [91], TAORF (cpg0017-rohban-pathways) [113]. Treatment: Chemical compounds, or ORF overexpression. Controls: number of control wells. Treatments: number of distinct treatments.  $N_r$ : median number of replicates per treatment.

Dataset	Treatment type	Cell line	Controls	Treatments	$N_r$
CDRP-bio	Chemical	U2OS	3,528	2,239	8
LINC	Chemical	A549	26,572	1,458	5
LUAD	Genetic	A549	320	593	8
TAORF	Genetic	U2OS	120	323	5

are measured as the difference between the CellProfiler representations to the autoencoder output predictions, and define the anomaly-representations of treated wells (Fig. 4.1C). We estimated the in-distribution reconstructed error according to the control wells that were not included for training. The anomaly-representation of a treatment was defined as the deviation of the treated well’s reconstruction error, one feature at a time, in respect to the in-distribution controls’ reconstruction errors (Fig. 4.1D). High reconstruction errors indicate that the in-distribution autoencoder was not able to effectively reconstruct the wells’ profiles, suggesting that these treatments alter the morphological and intracellular organization in relation to control cells. These anomaly-representations can be used for downstream analyses such as hit identification in screening, assessment of treatment reproducibility and MoA identification [83, 88] (Fig. 4.1E). Importantly, the reconstruction error of each biologically-interpretable feature provides a direct “mechanistic” explanation, thus benefiting from transparent interpretability of hand-crafted features along with deep-learning capitalization of non-linearity and the wealth of control data.

## 4.2.2 Anomaly-based Representations Enhance Reproducibility

Reproducibility, the extent to which a phenotype is replicated under the same experimental treatment, is a critical requirement in any screening application, where only a small fraction of treatments are followed-up with further comprehensive experimental validations (e.g., in drug discovery [114]). In high-content cell profiling, reproducibility serves as a measurement for the consistency of the experimental protocol and the efficacy of the profile’s representation and is used to exclude low-reproducible compounds from downstream analyses. We evaluated the reproducibility of our anomaly-representation in comparison to the CellProfiler representations in replicates of the same treatment across different plates. Reproducibility was measured by standardization of the well’s replicate-level profiles per plate, followed by measurement of the “Percent Replicating” score [83, 88, 91], the fraction of reproducible treatments, where a compound is deemed reproducible if its median pairwise profile correlation across replicates (“Replicate Correlation”), exceeds a threshold percentile of the pairwise correlation of random pairs of replicates across treatments (“Random Pairs Correlation”) (Fig. 4.2A-B). Following [88], we considered this threshold to be the 90<sup>th</sup> percentile of the random pairs’ distribution. We measured reproducibility of the anomaly- and of the CellProfiler-representations on four publicly available Cell Painting datasets from the Cell



**Figure 4.2: Anomaly-based representations are more reproducible.** (A) Cartoon depicting the reproducibility determination. A treatment is defined as reproducible if the median pairwise correlation of its replicates (Replicate Correlation) is higher than the 90<sup>th</sup> percentile of the pairwise correlation. (B) Cartoon depicting the Percent Replicating score. The distribution of Replicate Correlations (green) versus the distribution of Random Pairs Correlations (black). The dashed red vertical line defines the reproducibility threshold of 90% of the random pairs distribution - a well to the right of this line (green cell icon) is defined as reproducible, and the fraction of reproducible treatments determines the Percent Replicating score. (C) Percent Replicating scores across datasets for the anomaly-based (left) and the CellProfiler-based representations (right). Distribution of Replicate Correlations (green - anomaly-based, red - CellProfiler-based). Distribution of Random Pairs Correlations (gray), zero correlation (dashed gray vertical line), reproducibility threshold (dashed red vertical line). (D) Venn diagram showing the number of reproducible treatments exclusive to the anomaly-based (green) or CellProfiler-based (orange) representations, and common to both representations (yellow).

**Table 4.2: Reproducibility results.** Percent Replicating score for the CellProfiler (left column) versus the anomaly-based (middle column) representations. The union of treatments found reproducible by either representation is shown in the right column ( $A \cup C$ ).

Dataset	CellProfiler (%)	Anomaly (%)	$A \cup C$ (%)
CDRP-bio	201 (8.9)	418 (18.66)	429 (19.16)
LINCS	1118 (76.68)	1259 (86.35)	1329 (91.15)
LUAD	68 (62.05)	438 (73.86)	455 (76.72)
TAORF	189 (58.51)	180 (55.72)	221 (68.42)

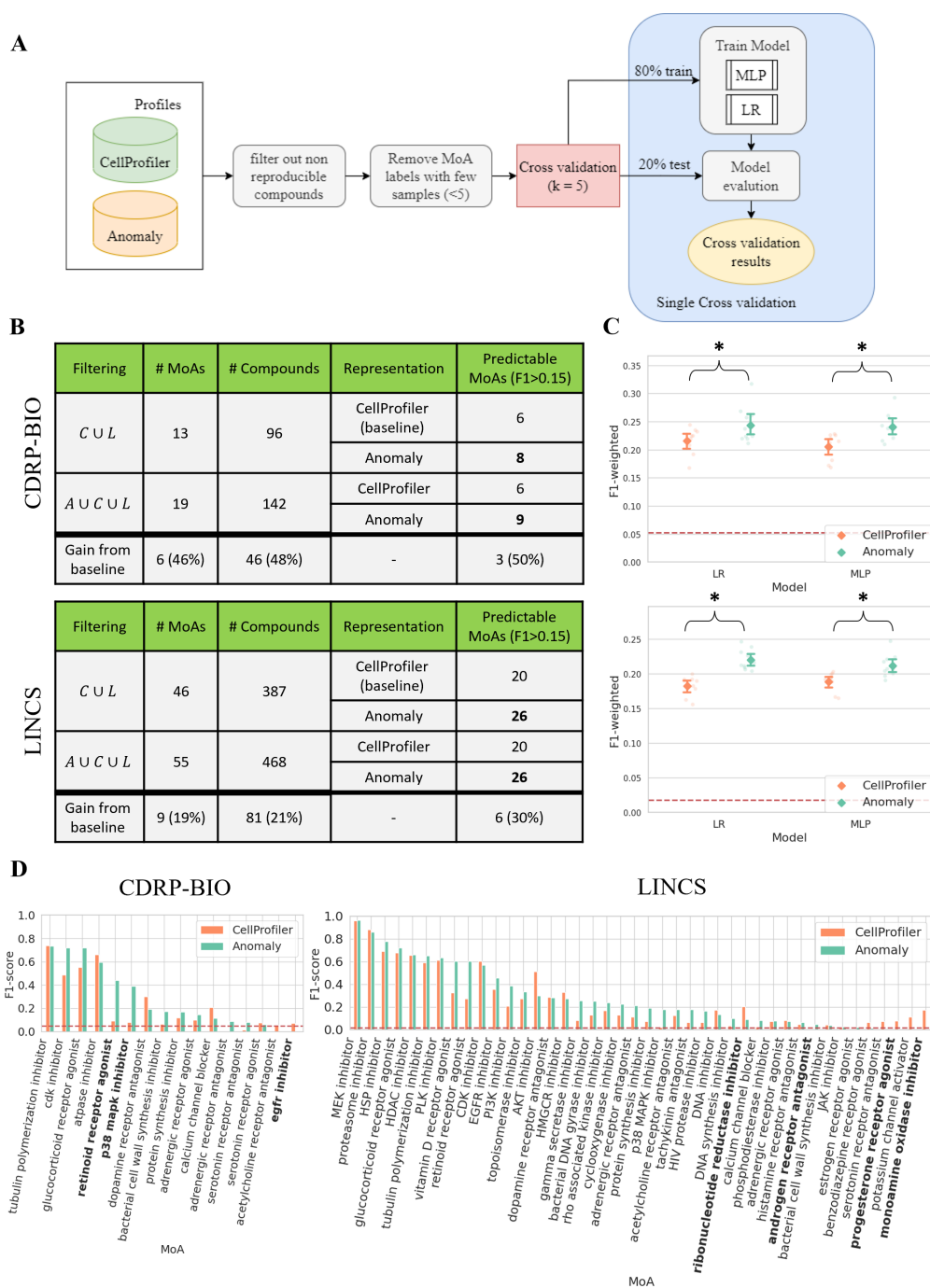
Painting Gallery [115] (Table 4.1): two compound screens (CDRP-bio [80], LINCS [83]) and two open reading frames (ORF) overexpression screens (LUAD [91], TAORF [113]); TAORF contains overexpression of WT cDNAs while LUAD contains overexpression of both WT and genetic variants. The median number of replicates per treatment ( $N_r$ ) in each dataset ranges between 5 to 8.

The anomaly-based representations were more reproducible than the CellProfiler representations in three out of the four datasets (Fig. 4.2C and Table 4.2): percent replicating score of 18.6% versus 8.9% correspondingly (i.e., a 2.09 fold increase) in the CDRP-bio dataset, 86.3% versus 76.7% (i.e., a 1.12 fold increase) in the LINCS dataset, and 73.8% versus 62.05% (i.e., a 1.19 fold increase) in the LUAD dataset. For TAORF, the smallest of the inspected datasets, CellProfiler representations reproduced 58.51% of the ORF overexpression versus 55.72% reproduced by the anomaly-based representation (i.e., a 1.05 fold increase in favor of CellProfiler). The deterioration in reproducibility could be related to the limited number of control wells in the TAORF dataset, with only 60 (out of 120) control wells used for training the in-distribution autoencoder. For comparison, the CDRP-bio dataset has 3,528 control wells (Table 4.1). Systematic assessment of the percent replicating score as a function of the number of control wells verified that increased numbers of control wells enhance the anomaly-based representations reproducibility, probably by learning a more accurate representation of control well's in-distribution (Supplementary Fig. 4.1). Low performance was observed for TAORF and CDRP-bio datasets on previous research and was hypothesized to derive from poorer technical data quality [88]. Together with the low number of training samples, that may lead to inaccurate in-distribution learning. The distributions of Replicate Correlations were lower for the anomaly-based representation, regardless of being more reproducible in comparison to CellProfiler representations, because their corresponding Random Pairs Correlations were more concentrated around 0 (Fig. 4.2C). These low correlations among random replicates suggested that the anomaly-based representations were less sensitive to experimental batch effects that may lead to spurious correlations. We next evaluated what fraction of the reproducible treatments were common and what fraction was distinct for the anomaly-based and CellProfiler representations (Fig. 4.2D). Most of the reproducible treatments were consistently identified in both representations, with larger numbers of reproducible treatments exclusively found by the anomaly-based representation for all datasets excluding TAORF. The union of the reproducible treatments found by either representation leads to a major improvement in the numbers of reproducible treatments that can be used for downstream analyses such as identifying their Mechanism of Action: 80% more in CDRP-bio,

25% in LINCS, 24% in LUAD, and 42% in TAORF. Finally, we evaluated whether the anomaly-based representations also provided complementary information to the mRNA expression of 978 genes (called the *L1000* profile). Analysis of the complementary of all three representations showed that the anomaly-based representations are complementary to both CellProfiler and L1000 (Supplementary Fig. 4.2). Altogether, these results indicate that anomaly-based representations encapsulate complementary information with respect to the CellProfiler representations that can lead to the availability of more reproducible treatments for followup investigations.

### 4.2.3 Anomaly-based Representations Enhance Mechanism of Action Identification

Linking a compound to its mechanism of action (MoA) is an important application of image-based phenotypic profiling [87, 83, 88, 89, 116]. We evaluated the ability to identify the MoA of compounds using anomaly-based representations in comparison to the CellProfiler representations. CDRP-bio and LINCS include compounds with known MoA and were thus used for this evaluation. We followed a previously described workflow to benchmark MoA identification on these datasets [83, 88] (Fig. 4.3A). First, we included compounds that were found reproducible by one of the anomaly-based, CellProfiler, or L1000 representations. Second, we excluded MoAs with fewer than 5 reproducible compounds linked to them. The complementarity between anomaly-based, CellProfiler and L1000 representations leads to inclusion of more compounds that leads to inclusion of more MoAs with sufficient numbers of treatments. Third, we evaluated compounds' MoAs predictions for different representations, with a 5-fold cross-validation using Logistic Regression (LR) and Multi Layer Perceptron (MLP) classifiers. This process was repeated 10 times for robustness analysis. The comparison between the anomaly-based and the CellProfiler-based representations was performed using two inclusion criteria for selecting reproducible compounds to train MoA classifiers: according to (1) CellProfiler and L1000 (*CUL*), or according to (2) Anomaly, CellProfiler, and L1000 (*AUCUL*). Inclusion of reproducible compounds determined by the anomaly-based representations (*AUCUL*) increased the number of MoAs (46% more for CDRP-bio and 19% more for LINCS), and increased the total number of reproducible compounds (48% for CDRP-bio and 21% for LINCS) (Fig. 4.3B). For each of the two inclusion criteria subsets the anomaly-based representations exhibited superior performance compared to the CellProfiler baseline for both the LR and the MLP classification models, with an average weighted F1-score of 0.244 versus 0.216 for the best model trained with anomaly-based versus CellProfiler representations correspondingly for the CDRP-bio dataset, and 0.22 versus 0.188 for the LINCS dataset (Fig. 4.3C). Specifically, the anomaly-based representations lead to prominent improvements in the CDRP-bio mechanisms of actions retinoid receptor agonist ( $\Delta$ F1-score = 0.35), p38 MAPK inhibitor ( $\Delta$ F1-score = 0.31) and CDK inhibitor ( $\Delta$ F1-score = 0.23), and in the LINCS MoAs for retinoid receptor agonist ( $\Delta$ F1-score = 0.33), vitamin D receptor agonist ( $\Delta$ F1-score = 0.28), and in PI3K inhibitor ( $\Delta$ F1-score = 0.18) (Fig. 4.3D). Additionally, anomaly-based representations were also found to be complementary to the genetic L1000 profiles, showing improved results when concatenated together compared to using each one independently (Supplementary Fig. 4.3)). Altogether, anomaly-based representations improve MoA classification



**Figure 4.3: Anomaly-based representations improve MoA classification.** (A) MoA classification workflow. Left-to-right: inclusion of compounds that met our reproducibility criteria (for either the CellProfiler or the Anomaly-based representations), followed by exclusion of MoAs with  $< 5$  compounds attributed to them. Training machine learning models using the anomaly-based versus the CellProfiler-based representations with cross-validation. (B) MoA classification results for anomaly-based versus CellProfiler-based representations, using the reproducible inclusion criteria with ( $A \cup C \cup L$ ) or without ( $C \cup L$ ) the anomaly-based representations. (C) F1-scores of anomaly-based (green) versus the CellProfiler-based (orange) representations trained on MLP and LR models for CDRP-bio (cpg0012-wawer-bioactivecompoundprofiling) and LINCS (cpg0004-lincs) datasets. Each dot represents an experiment and bars indicate confidence intervals. Red dashed line indicates the F1 random score. \* - statistically significant ( $p$ -value  $< 0.05$ ) by Welsh t-test. P-values for CDRP-bio were 0.042 (LR) and 0.003 (MLP) and for LINCS 1.5e-10 (LR) and 0.003 (MLP) (D) MoA-specific F1 scores using the (better performing) LR model. Bold indicates MoAs that would not be included without reproducible compounds according to the anomaly-based representations. F1-scores lower than the random score for both representations (1/19 MoAs in CDRP-bio, and 9/55 MoAs in LINCS) were excluded from the figure to improve clarity.

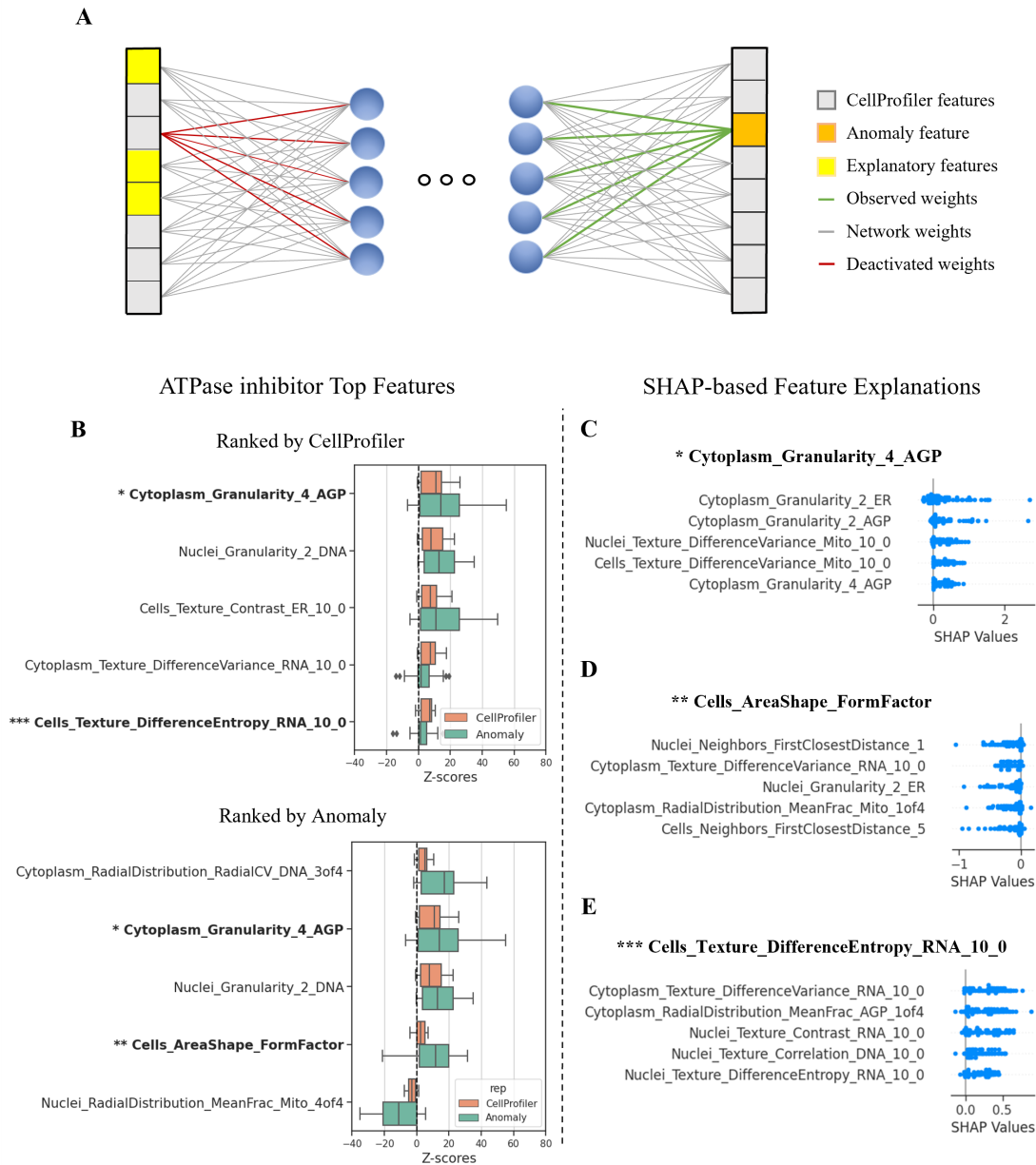


by both increasing the number of reproducible compounds available for MoA classification and by encoding more discriminative information than the CellProfiler-based representations.

### 4.3 Anomaly-based Representations are Interpretable

Understanding the underlying reasons behind a treatment's hit identification is crucial for meaningful interpretation and subsequent decision-making. The CellProfiler hand-crafted features are linked to specific morphological phenotypes, and thus treatment-induced alterations of many of these features can be biologically interpreted and investigated. Interpretation of the anomalous features is more challenging because an anomalous (i.e., poorly reconstructed) feature can be “caused” by a combination of subtle alterations in several inter-dependent input features (Fig. 4.4A). To interpret our anomaly-based representation we applied a recent unsupervised extension of the widely used SHAP (SHapley Additive exPlanations) [117] that was designed to explain anomalies identified by an autoencoder [118]. SHAP assigns importance values to features based on their additive contribution to the model output, and its extension to autoencoder-based anomalies isolates the input features contributing to the high reconstruction errors by treating each feature as a separate reconstruction task (Fig. 4.4A). We focused on the interpretation of the ATPase inhibitor MoA, which was one of the top predicted MoA classes in the CDRP-bio dataset (Fig. 4.3D). When pooling all ATPase inhibitor treatments we found that the CellProfiler feature “Cytoplasm\_Granularity\_AGP\_4” was ranked first in terms of its deviation from the control, and was ranked second in the anomaly-based representation (Fig. 4.4B, single asterisk). Granularity measures the signal lost with iterative erosions relative to the total signal and can be hard to directly interpret. The autoencoder-based anomaly SHAP explanation for “Cytoplasm\_Granularity\_AGP\_4” included the “adjacent” Granularity feature, “Cytoplasm\_Granularity\_AGP\_3”. This result aligned with a recent study [119] reporting that genetically disrupting the Vacuolar ATPase (which is a specific ATPase) caused a specific decrease in the first Granularity feature in the Golgi and plasma membrane channel (the most similar channel to the F-actin, Golgi, and plasma membrane (AGP) channel in the CDRP-bio dataset), and a concomitant increase in other Granularity features. Our feature selection excluded some AGP Granularity features thus not allowing us to fully replicate this result. To test this link between granularity features, we retrained the in-distribution autoencoder using the full feature set, without selection. This analysis verified that the granularity phenomenon is reproduced for the ATPase inhibitor MoA in both the CellProfiler and the anomaly-based representations (Supplementary Fig. 4.4). Though none of the compounds in the ATPase inhibitor MoA class in CDRP-bio was annotated to directly target the Vacuolar ATPase, the similarity of our observations can easily be explained by either perturbation causing a general disruption of intracellular acidification and therefore similar observable phenotypes.

A feature with a high reconstruction error but an unperturbed CellProfiler value implies that the in-distribution autoencoder reconstruction was hampered by the combined effect of alteration in other features. An example is “Cells\_AreaShape\_FormFactor” (Fig. 4.4B, double asterisk). The



**Figure 4.4: MoA interpretation.** (A) Illustration of the explanation process of the anomalous features (orange on the right). This anomaly is explained by the combined alteration of multiple input features (yellow on the left). A group of well-level profiles of interest are passed through the network, and the features exhibiting the highest reconstruction errors are identified. For each investigated feature (orange square in output), the weights leading to its activation (green lines) are activated and the weights going from its CellProfiler representation in the input are deactivated (red lines). The well-level profiles are reintroduced into the network and the features that led to high reconstruction errors are found by using the autoencoder-based anomaly SHAP. These explanations can be pooled according to a treatment or MoA to provide the corresponding explanation. (B) Distributions of ATPase inhibitor top five features' z-scores for the CellProfiler (orange) and anomaly-based representations (green) ranked according to the CellProfiler median z-scores (top) and by anomaly median z-scores (bottom) in the CDRP-bio dataset. Features in bold are analyzed using the autoencoder-based anomaly SHAP in the following panels. (C-E) Explanations for selected (see text for justification) altered features in the ATPase inhibitor MoA in the CDRP-bio dataset. Each dot represents a replicate well, and the x-axis represents the SHAP values contributing to the errors. Negative values indicate an inverse relationship between the inspected feature and the input feature compared to the relationship in the control population.



FormFactor feature measures the roundness of objects by calculating the ratio between the object area and perimeter, i.e., larger values indicate more circular objects. The autoencoder-based anomaly SHAP explanation of this feature included two features that measure the cell's distance from neighboring cells ("Neighbors\_FirstClosestDistance" features), with higher distances in the input leading to lower cell roundness in the reconstruction (Fig. 4D). Typically, distance from neighbors is a measure of local cell sparsity, and cells at sparse environments have more space to spread and thus may have lower FormFactor values. It is intriguing that this relationship is affected in this MoA and further exploration could be done to test how frequently this relationship is broken across MoA classes and whether this is correlated with compound toxicity as AreaShape features are known to play an important role in predicting Cell Health readouts [120].

A feature with a low reconstruction error but a perturbed CellProfiler value implies that the inter-relations learned in the in-distribution autoencoder were maintained and thus the deviation of the feature can be properly reconstructed by using alterations of other features. An example is "Cells\_Texture\_DifferenceEntropy\_RNA\_10\_0" (Fig. 4.4B, triple asterisk). This feature's autoencoder-based anomaly SHAP explanation identified positive relations with other texture features and with the radial distribution mean fraction in the cytoplasm AGP channel (Fig. 4.4E). Texture features are highly represented in Cell Painting datasets, but are not human-interpretable. It is possible that exploring interpretable SHAP explanations rather than uninterpretable features provides a welcome alternative approach to understanding the biology behind low interpretable CellProfiler features.

Cumulatively, our results indicate that autoencoder-based anomaly SHAP explanation of anomaly-based representations can reveal specific biological-interpretable dependencies between features that break following a treatment.

## 4.4 Discussion

We take advantage of the abundance of control wells, to formulate the high-content image-based cell profiling as an anomaly-detection problem. Our anomaly-based representations learn the inter-feature complex dependencies of the population of "normal" (i.e., unperturbed) cells by minimizing the control reconstruction errors under the premise that perturbations in the cells' organization will lead to less successful reconstruction due to altered inter-feature dependencies. Such representations hit a sweet-spot in terms of the inherent tradeoff between performance and interpretability. First, our anomaly-based representations model the dependencies between features, surpassing traditional methods of analyzing CellProfiler-representations where features are considered independently in reproducibility (Fig. 4.2) and MoA classification (Fig. 4.3). The low correlations among random replicates (Fig. 4.2C) suggests that these representations implicitly mitigate batch effects, a known confounder of cell profiling, meriting the future application of anomaly-based representations for batch correction [121]. Second, optimizing representations

to capture these inter-feature dependencies provide a complementary readout that can identify perturbations that extensively alter these dependencies, without necessarily sufficiently altering each of the individual features (Figs. 4.2,4.3). For example, several positively correlated features may lead to an anomaly due to a marginal increase of some features along with a marginal decrease of other features. These subtle phenotypes are not sufficiently profound to create a reproducible phenotype, without encoding the deviation of the non-linear convoluted dependencies. Third, using the CellProfiler-derived features as the starting point for our anomaly-representations make the latter more interpretable in respect to most deep learning “black-box” representations. Specifically, using unsupervised explainability we were able to extract biologically-meaningful dependencies that deviated in response to a perturbation (Fig. 4.4).

Weakly supervised representation-learning use experimental labels, such as the treatment label, to “guide” their representations such that replicates of the same label are encoded close to one another in the latent space and different labels are encoded far apart [95, 96, 97, 98, 99, 100]. Beyond optimizing representations that are hard to biologically-interpret, experiment treatment-based weak supervision may lead to undesired consequences where the representations of treatments that lead to a similar phenotype are pushed away from one another in the latent space because they do not share the same label, possibly even pushing one representation closer to the control (Supplementary Figure 4.5A). In turn, representations with reduced cross-treatment phenotypic similarity may induce errors in downstream analyses, especially in unsupervised interpretation of biological function such as lead optimization [122]. In contrast, our anomaly-based method is self-supervised, i.e., does not use treatment labels (or any other assumptions on the underlying data) to guide the representation, rather treatment profiles’ latent representations are encoded solely according to their deviation from the control in-distribution (Supplementary Fig. 4.5B).

The core idea of learning the in-distribution of control experiments followed by identifying anomalies with respect to this distribution is a general concept that can be adapted to applications in cell biology beyond high-content image-based cell profiling. For example, recent work suggested a form of anomaly detection to identify treatments leading to altered cell fate (apoptosis or mitosis) decision processes during live imaged cell competition assay by training a model for the prediction of “normal” cell fate behavior and a discriminator network to determine anomalies deviating from the expected cell decision process [123]. Of course, any anomaly-based representation requires sufficient control replicates to properly model the in-distribution of the data (see Supplementary Figure 4.1). In applications where there are no controls, but sufficient data, anomaly-based representations can still be applied by learning the in-distribution of one experimental condition and then modeling the reconstruction errors of the other experimental conditions with respect to the in-distribution reconstruction errors. For example, in single-cell spatial multiplexed proteomics applied to clinical data, each patient has many cells, and thus we can learn the in-distribution of the protein expression across cells from one disease state and apply it to patients from other disease states. Our application involved first deriving pre-defined features from the raw image and then applying anomaly detection on the tabular representation. A similar approach can be applied directly to the raw image data using convolutional neural networks (e.g., [124]). Learning

the in-distribution directly from the raw images avoids segmentation errors and can unbiasedly encode spatial relations that were not measured by the CellProfiler features, but suffer from poor interpretability. Imaged-based anomaly detection can also have applications in other domains such as single-cell spatial omics.

## 4.5 Methods

### 4.5.1 Datasets

All datasets were created at the Broad Institute and previously published. Two datasets had chemical perturbations and two had genetic perturbations (Table 4.1). For each dataset two distinct readouts were recorded: gene expression (GE) profiles and morphological profiles (Cell Painting). Each dataset was acquired by plating cells in two sets of identical plates, perturbed identically in the same laboratory. One of these sets was used to measure GE and the other set was imaged to measure morphology. The morphological profiles were captured using the Cell Painting assay [85]. Cell painting is a microscopy-based assay that acquires five fluorescence channels labeling of the actin cytoskeleton, Golgi apparatus, plasma membrane, nucleus, endoplasmic reticulum, mitochondria, nucleoli and cytoplasmic RNA. Acquired microscopy images were processed using the CellProfiler software [86] to extract 1,783 features of each cell's morphology such as shape, intensity and texture statistics, that were aggregated (population-averaged) to create per-well profiles. The features are measured in three cell regions – nucleus, whole-cell, and cytoplasm (difference between whole-cell area and nuclei area) . The GE profiles were acquired using the L1000 assay [125], that provides high-throughput measurement of mRNA levels for 978 genes, roughly covering 82% of the transcriptional variance across the entire genome. Cell Painting profiles were used for anomaly detection representations. GE profiles were used as additional input for selecting reproducible treatments and for MoA classification. We downloaded all metadata-augmented per-well aggregated Cell Painting datasets from the Cell Painting Gallery (CPG) (<https://registry.opendata.aws/cellpainting-gallery/>) [115]. We downloaded the preprocessed profiles and performed normalization and feature selection in the training process to avoid data leakage during normalization (See Data preprocessing section). GE profiles were also downloaded from the CPG [115].

### 4.5.2 Data Split Approach

Each experimental plate had multiple control replicates and many treated wells without per-well replicates (i.e., one replicate per treatment in a well). Per plate, the control wells were randomly split to 40% : 10% : 50% between train, validation and test, correspondingly. Thus, each plate's wells were split to four subgroups of wells: (1) training controls, (2) validation controls, (3) test controls, and (4) treated wells for evaluation at inference.

### 4.5.3 Data Preprocessing

Per-plate feature-wise standardization was implemented on all well-level samples using the aggregated training controls population. Normalization was exclusively conducted on the training set to prevent any leakage of the test data distribution during the normalization process. Whole-plate normalization was performed after training (See “Measuring reproducibility” section). Feature selection was performed using the Pycytominer software [126], removing features with high ratio of null values (`null_threshold = 0.05`), high correlation (`corr_threshold = 0.9`), and low variance (`freq_cut = 0.05`, `unique_cut = 0.01`), all with the default parameters as set by Pycytominer. Following feature selection, the number of features dropped from 1,700 to a range of 300-600 for all datasets.

### 4.5.4 Anomaly Detection

The autoencoder architecture consisted of a large encoder and a flat decoder. Three-layer encoder, with layer sizes 256-128-64, where each layer contains a Relu activation layer. The latent dimension was of size 16. A 1-layer decoder transformed the latent representation back to the input size. The loss function was the mean square error between the reconstructed output and the corresponding observed CellProfiler representation, with an additional l2 regularization loss to restrict the output amplitude. Each autoencoder (one per dataset) was trained until convergence or a maximum of 300 epochs with batch size of 32. Hyperparameter optimization was applied for each dataset with a learning rate range of  $1e-4$  to  $1e-2$ , output features l2 regularization range of  $1e-5$  to  $1e-2$ , and dropout ratio of 0 to 0.2. Total time for training each of the datasets, including hyperparameter optimization, was up to one hour on the Nvidia RTX1080 GPU.

At inference, the trained autoencoders calculated the feature-wise predictions’ reconstruction error for each well called the anomaly-based representation. All treatment replicates were z-score normalized according to the test control wells. These steps resulted in a vector of feature-wise z-scores for each treatment replicate.

### 4.5.5 Measuring Reproducibility

Whole-plate normalization was applied based on all replicate-level z-scores, including the control and the treatment replicates, because it was the optimal setting for reproducibility, in agreement with previous studies [88, 119]. For all representations (CellProfiler-based, Anomaly-based, L1000), reproducibility was measured according to the fraction of reproducible treatments, a measurement called the “Percent Replicating” score [83, 88, 91]. Briefly, for each treatment, we first calculated the mean Pearson correlation coefficient of each pair of replicates profiles. Second, we compared the median replicate pairs correlation of each treatment to a null distribution containing correlations of all random pairs of treatments. Third, for robustness, we repeated the previous step for five times,

altogether generating a null distribution containing  $5 * \binom{n_{treatments}}{2}$ . Last, following [88], we defined a treatment as reproducible if its mean Replicate Correlation was above the 90<sup>th</sup> percentile of the null distribution. The LINCS had multiple doses per compound, and reproducibility was measured based on replicates of the highest dose for each compound.

#### 4.5.6 MoA Classification

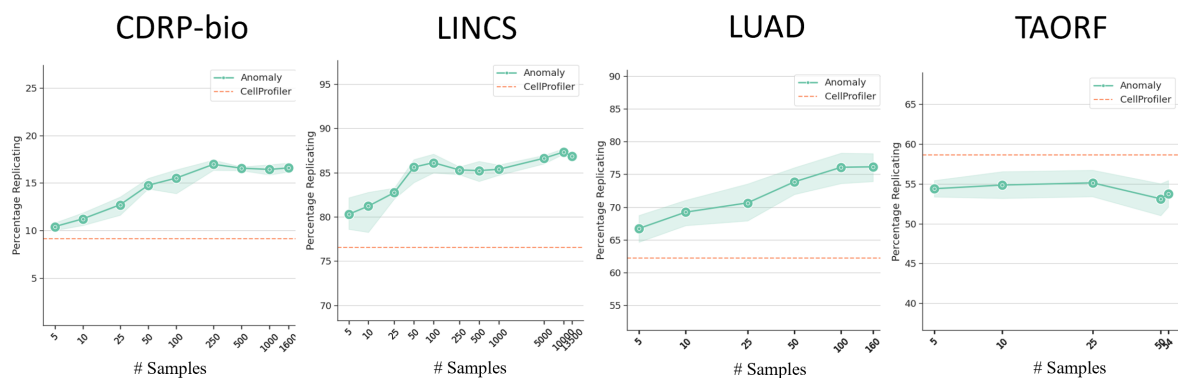
We compared MoA classification based on CellProfiler-based and anomaly-based representations for the CDRP-bio and LINCS datasets that included compounds with known MoA (Supplementary Fig. 4.6). First, for each representation, each treatment was represented by its mean profile across plate-normalized replicates. Second, following [88], we selected reproducible treatments according to two inclusion criteria: (1) Reproducible treatments according to the CellProfiler or L1000 representations, or (2) Reproducibility according to the Anomaly, CellProfiler, or L1000 representations. Third, we excluded all MoAs with fewer than five reproducible treatments. All treatments' features were scaled to a range of [0,1] and were used to train a Logistic Regression (LR) and a Multi Layer Perceptron (MLP) MoA classifiers. Each model was applied for prediction of MoA labels using CellProfiler-based and anomaly-based representations independently to compare the representations. We performed stratified nested k-fold cross-validation ( $k = 5$ ) to evaluate the classification performance. The MLP classifier architecture was made of two layers with a Relu activation layer between them. Hidden layer sizes, regularization strength, and learning rates were optimized per training fold. LR was optimized with different regularization parameters per training fold. MoAs vary in compound numbers, ranging from five to 25 (Supplementary Fig. 4.6). To address data imbalance, models underwent oversampling to align class sizes with the majority class in the training set. The predictions were evaluated using the F1-score metric. A naive random F1 score for classes was calculated for baseline. Each classification experiment was repeated 10 times for robustness. The statistical significance of the difference between CellProfiler and Anomaly-based representations was calculated using the Welsh t-test, a two-sample test used to test if two populations have different means. The test was performed with  $N=10$ , the number of experiments per setting.

#### 4.5.7 Autoencoder-based Anomaly SHAP Explanations

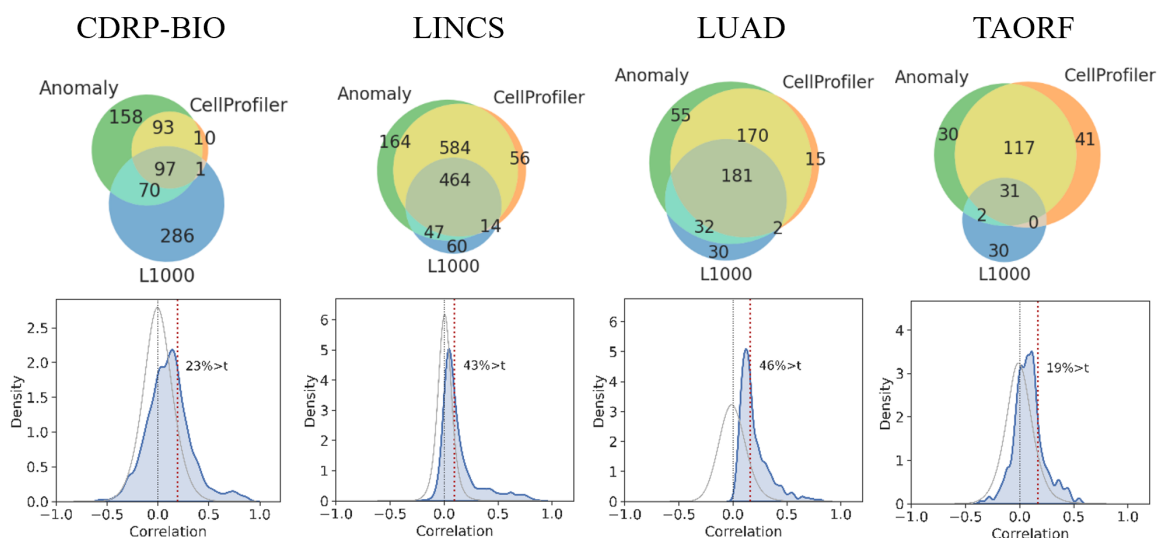
To interpret what were the inter-feature associations that deviated following a treatment we used an extended version of the classic Shapley additive explanation (SHAP) method [117] that was designed to explain anomalies identified by an autoencoder [118]. For a treatment or MoA of interest, the autoencoder-based anomaly SHAP explanations were obtained independently for each anomalous feature by calculating the autoencoder reconstructed output values of that feature and then the SHAP values of the input features in relation to the output feature at test. First, the most anomalous features were identified according to the reconstruction errors. Second, given an anomalous feature, the weights associated with this feature in the first layer were deactivated

and the well-level replicate profiles were re-introduced to the network. The output values of the inspected feature were analyzed using kernel SHAP to compute the SHAP values of the input features, i.e. the importance of each input feature in predicting the anomalous feature. Kernel SHAP is a model-agnostic approximation of SHAP values via weighted linear regression. The Kernel SHAP estimates the additive value of features in the input space for a subset of samples of interest. The importance of an input feature is determined by the change in the model’s output when that input feature is set to “missing”. The “missing” values are replaced with values values taken from a reference dataset that reflects the distribution of the general data. We used samples from the control test set as the reference dataset for kernel SHAP approximation.

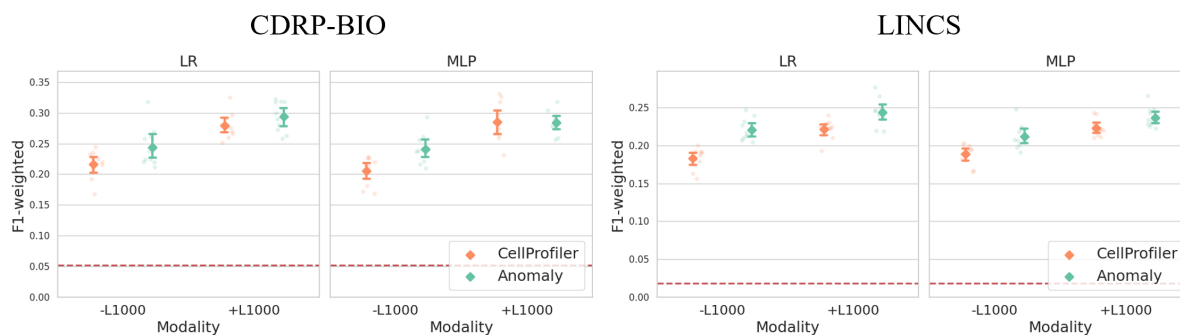
## 4.6 Supplementary Information



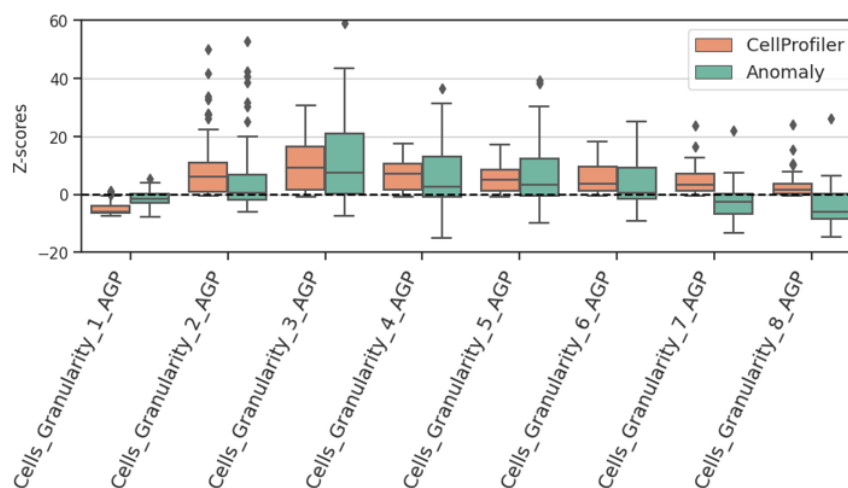
**Supplementary Figure 4.1:** Percent Replicating score as a function of the number of control wells used to train the in-distribution autoencoder. Dashed orange line - the percent replicating score of the CellProfiler representations. Green data points and shade show the percent replicating score mean and the standard deviation of the anomaly-based representations over 10 independent experiments.



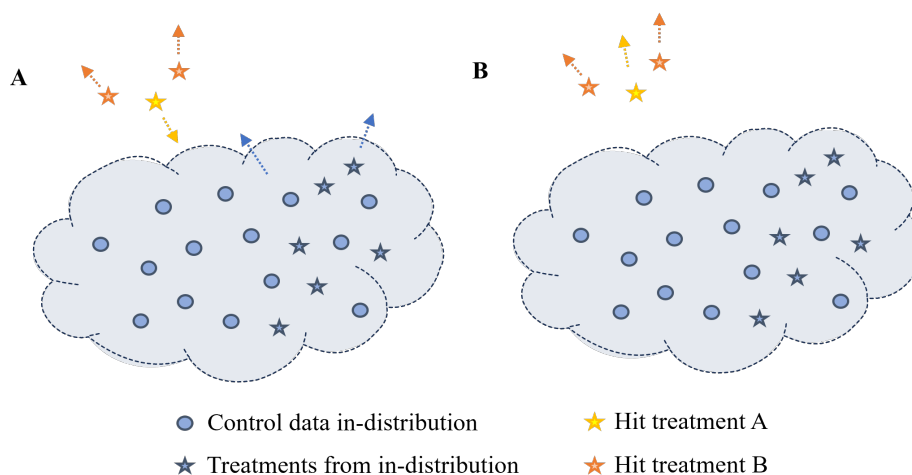
**Supplementary Figure 4.2:** Analysis of the complementary information shared between the anomaly-based, the CellProfiler, and the gene expression representations (L1000). **Top:** Venn diagram showing the number of reproducible treatments exclusive to the anomaly-based (green), CellProfiler-based (orange), and L1000-based (blue) representations. Treatments found reproducible by multiple representations are shown by the intersection of the different circles. **Below:** Percent Replicating scores for the L1000-based representations. Distribution of Replicate Correlations (blue), Random Pairs Correlations (gray), reproducibility threshold (dashed red vertical line).



**Supplementary Figure 4.3:** Genetic expression representations (L1000) complement the CellProfiler (orange) and anomaly-based (green) representations for MoA classification. Logistic Regression (LR) and Multi Layer Perceptron (MLP) MoA classification F1 performance on CellProfiler versus anomaly-based representations without (-L1000) or with (+L1000) the concatenation of the L1000 representation. Each dot represents a unique experiment and bars indicate 95% confidence intervals. Red dashed line indicates the F1 random score.

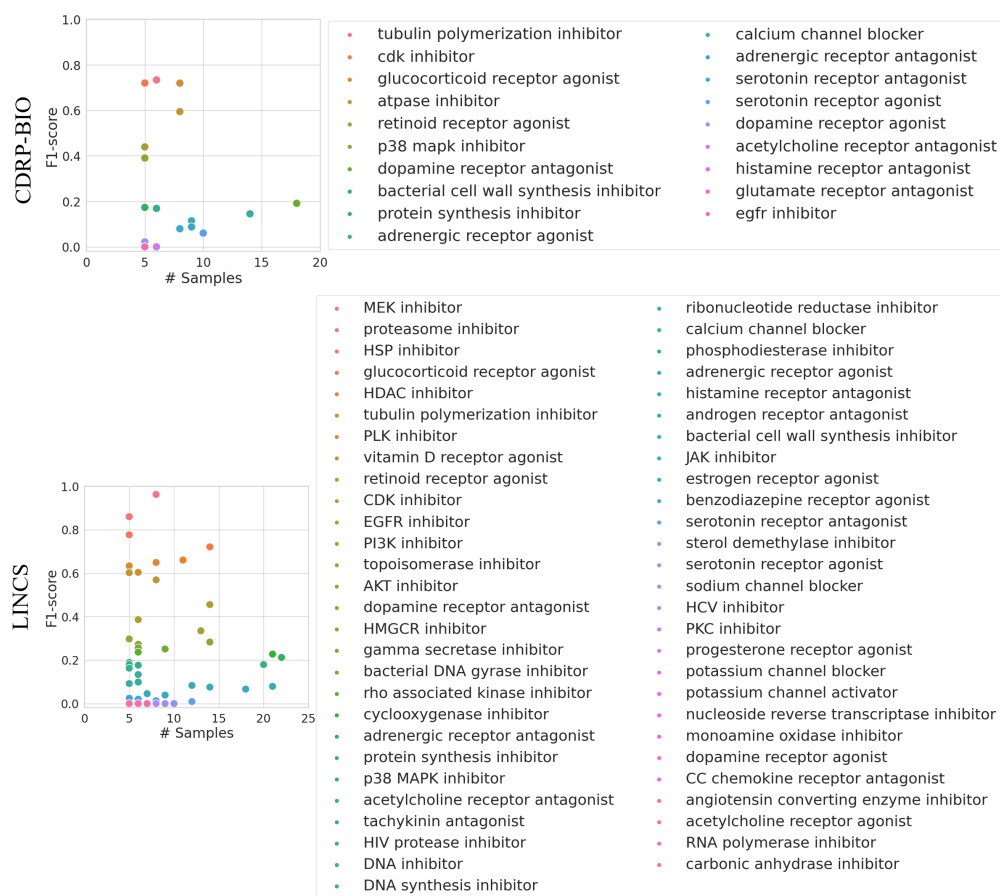


**Supplementary Figure 4.4:** Granularity analysis in the AGP channel for the MoA of ATPase inhibitor in the CDRP-bio dataset. Feature z-scores distributions for CellProfiler (orange) and anomaly-based (green) representations.



**Supplementary Figure 4.5:** Illustration of potential caveats of using weakly supervised (e.g., using the treatment as a weak label) (A) versus anomaly-based (B) representations. (A) A treatment representation (yellow) is implicitly becoming more similar to the control because it was guided away from another treatment (orange). (B) Anomaly-based representations are guided away from the control.





**Supplementary Figure 4.6:** Scatter plot indicating the F1 score as a function of the number of treatments for each MoA in the CDRP-bio and LINCS dataset. The F1 scores were obtained with the better-performing LR model trained using the anomaly-based representations.

#### 4.6.1 Funding and Acknowledgments

This research was supported by the Israel Science Foundation (ISF, grantNo. 2516/21, to AZ), the Israel Ministry of Science and Technology (MOST, to AZ), by the Center for Open Bioimage Analysis (COBA) and the National Institute of General Medical Sciences P41 GM135019 (EW). We thank Hillel Bar-Gera for his support and mentorship.

## Final Remarks

This dissertation has explored the intersection of diverse disciplines through the lens of deep learning, with a focus on advancing the state-of-the-art in three domains: interpretability of deep neural networks, signal processing for ultrasonic echo detection, and anomaly detection in high-throughput phenotypic screening. Each chapter has presented novel methodologies and approaches tailored to address specific challenges within its respective domain.

While each chapter diverged in its foundational principles, they all underscored the importance of relying on prior knowledge of the problem domain to drive meaningful progress. From probabilistic interpretations in the first chapter to the incorporation of physical principles in the second chapter and the utilization of biological experiment structures in the third chapter, these works exemplify the necessity of grounding deep learning advancements within the context of specific domains. Collectively, these chapters underscore the versatility and potency of deep learning methodologies in tackling complex challenges across disparate fields.

# Bibliography

- [1]Mingxing Tan, Ruoming Pang, and Quoc V Le. “Efficientdet: Scalable and efficient object detection”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 10781–10790 (cit. on p. 19).
- [2]Andrew Tao, Karan Sapra, and Bryan Catanzaro. “Hierarchical multi-scale attention for semantic segmentation”. In: *arXiv preprint arXiv:2005.10821* (2020) (cit. on p. 19).
- [3]Zihang Dai, Hanxiao Liu, Quoc V Le, and Mingxing Tan. “CoAtNet: Marrying Convolution and Attention for All Data Sizes”. In: *arXiv preprint arXiv:2106.04803* (2021) (cit. on p. 19).
- [4]Arun Das and Paul Rad. “Opportunities and challenges in explainable artificial intelligence (xai): A survey”. In: *arXiv preprint arXiv:2006.11371* (2020) (cit. on pp. 19, 22).
- [5]Xuhong Li, Haoyi Xiong, Xingjian Li, Xuanyu Wu, Xiao Zhang, Ji Liu, Jiang Bian, and Dejing Dou. “Interpretable Deep Learning: Interpretation, Interpretability, Trustworthiness, and Beyond”. In: *arXiv preprint arXiv:2103.10689* (2021) (cit. on pp. 19, 22).
- [6]Jun Yuan, Changjian Chen, Weikai Yang, Mengchen Liu, Jiazhi Xia, and Shixia Liu. “A survey of visual analytics techniques for machine learning”. In: *Computational Visual Media* 7.1 (2021), pp. 3–36 (cit. on pp. 19, 22).
- [7]Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. “Feature visualization”. In: *Distill* 2.11 (2017), e7 (cit. on pp. 19, 22).
- [8]Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014) (cit. on pp. 19, 22, 35).
- [9]Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. “Grad-CAM: Visual explanations from deep networks via gradient-based localization”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 618–626 (cit. on pp. 19, 22, 44, 46).
- [10]Fred Hohman, Haekyu Park, Caleb Robinson, and Duen Horng Polo Chau. “Summit: Scaling deep learning interpretability by visualizing activation and attribution summarizations”. In: *IEEE transactions on visualization and computer graphics* 26.1 (2019), pp. 1096–1106 (cit. on pp. 19, 24).
- [11]David Bau, Jun-Yan Zhu, Hendrik Strobelt, Agata Lapedriza, Bolei Zhou, and Antonio Torralba. “Understanding the role of individual units in a deep neural network”. In: *Proceedings of the National Academy of Sciences* 117.48 (2020), pp. 30071–30078 (cit. on pp. 19, 23).
- [12]Michael Sedlmair, Miriah Meyer, and Tamara Munzner. “Design study methodology: Reflections from the trenches and the stacks”. In: *IEEE Transactions on Visualization and Computer Graphics* 18.12 (2012), pp. 2431–2440 (cit. on p. 19).

- [13]Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. “ImageNet large scale visual recognition challenge”. In: *International Journal of Computer Vision* 115.3 (2015), pp. 211–252 (cit. on pp. 20, 35).
- [14]Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. “Understanding deep learning requires rethinking generalization”. In: *arXiv preprint arXiv:1611.03530* (2016) (cit. on pp. 22, 37).
- [15]Dumitru Erhan, Aaron Courville, and Yoshua Bengio. “Understanding representations learned in deep architectures”. In: *Department dInformatique et Recherche Operationnelle, University of Montreal, QC, Canada, Tech. Rep* 1355.1 (2010) (cit. on p. 22).
- [16]Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. “Understanding neural networks through deep visualization”. In: *arXiv preprint arXiv:1506.06579* (2015) (cit. on p. 22).
- [17]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*. 2016, pp. 1135–1144 (cit. on pp. 22, 23).
- [18]Matthew D Zeiler and Rob Fergus. “Visualizing and understanding convolutional networks”. In: *European Conference on Computer Vision*. Springer. 2014, pp. 818–833 (cit. on p. 22).
- [19]Ruth Fong, Mandela Patrick, and Andrea Vedaldi. “Understanding deep networks via extremal perturbations and smooth masks”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 2950–2958 (cit. on pp. 22, 23).
- [20]Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. “Learning deep features for discriminative localization”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 2921–2929 (cit. on p. 22).
- [21]Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. “Smoothgrad: removing noise by adding noise”. In: *arXiv preprint arXiv:1706.03825* (2017) (cit. on p. 23).
- [22]Naman Bansal, Chirag Agarwal, and Anh Nguyen. “Sam: The sensitivity of attribution methods to hyperparameters”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 8673–8683 (cit. on p. 23).
- [23]Giang Nguyen, Daeyoung Kim, and Anh Nguyen. “The effectiveness of feature attribution methods and its correlation with automatic evaluation scores”. In: *arXiv preprint arXiv:2105.14944* (2021) (cit. on p. 23).
- [24]Chris Olah, Arvind Satyanarayan, Ian Johnson, Shan Carter, Ludwig Schubert, Katherine Ye, and Alexander Mordvintsev. “The building blocks of interpretability”. In: *Distill* 3.3 (2018), e10 (cit. on p. 23).
- [25]David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. “Network dissection: Quantifying interpretability of deep visual representations”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 6541–6549 (cit. on p. 23).
- [26]Gromit Yeuk-Yin Chan, Enrico Bertini, Luis Gustavo Nonato, Brian Barr, and Claudio T Silva. “Melody: Generating and Visualizing Machine Learning Model Summary to Understand Data and Classifiers Together”. In: *arXiv preprint arXiv:2007.10614* (2020) (cit. on p. 23).
- [27]Quanshi Zhang, Ying Nian Wu, and Song-Chun Zhu. “Interpretable convolutional neural networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 8827–8836 (cit. on p. 23).

- [28]Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K Su. “This looks like that: deep learning for interpretable image recognition”. In: *Advances in Neural Information Processing Systems*. 2019, pp. 8928–8939 (cit. on pp. 23, 24).
- [29]Mengchen Liu, Jiaxin Shi, Zhen Li, Chongxuan Li, Jun Zhu, and Shixia Liu. “Towards better analysis of deep convolutional neural networks”. In: *IEEE transactions on visualization and computer graphics* 23.1 (2016), pp. 91–100 (cit. on p. 24).
- [30]Nicholas D Socci, Daniel D Lee, and H Sebastian Seung. “The rectified Gaussian distribution”. In: *Advances in Neural Information Processing Systems*. 1998, pp. 350–356 (cit. on p. 25).
- [31]Gyemin Lee and Clayton Scott. “EM algorithms for multivariate Gaussian mixture models with truncated and censored data”. In: *Computational Statistics & Data Analysis* 56.9 (2012) (cit. on pp. 26, 27).
- [32]Olivier Cappé and Eric Moulines. “On-line Expectation–Maximization algorithm for latent data models”. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 71.3 (2009), pp. 593–613 (cit. on p. 26).
- [33]Lawrence R. Rabiner and B. H. Juang. “An introduction to hidden Markov models”. In: *IEEE ASSP Magazine* (1986), 4–15 (cit. on p. 26).
- [34]Richard J. Lipton and Robert E. Tarjan. “A separator theorem for planar graphs”. In: *SIAM Journal on Applied Mathematics* 36(2) (1979), 177–189 (cit. on p. 28).
- [35]Alan Julian Izenman. “Linear discriminant analysis”. In: *Modern Multivariate Statistical Techniques*. Springer, 2013, pp. 237–280 (cit. on p. 33).
- [36]G David Forney. “The Viterbi algorithm”. In: *Proceedings of the IEEE* 61.3 (1973), pp. 268–278 (cit. on p. 33).
- [37]Yann LeCun, Corinna Cortes, and CJ Burges. “MNIST handwritten digit database”. In: *AT&T Labs [Online]*. Available: <http://yann.lecun.com/exdb/mnist> 2 (2010) (cit. on p. 35).
- [38]Alex Krizhevsky, Geoffrey Hinton, et al. *Learning multiple layers of features from tiny images*. Tech. rep. Citeseer, 2009 (cit. on p. 35).
- [39]Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778 (cit. on p. 35).
- [40]Guang-Ming Zhang, Cheng-Zhong Zhang, and David M Harvey. “Sparse signal representation and its applications in ultrasonic NDE”. In: *Ultrasonics* 52.3 (2012), pp. 351–363 (cit. on pp. 48, 50, 52).
- [41]Tom Pialucha and Peter Cawley. “The detection of thin embedded layers using normal incidence ultrasound”. In: *Ultrasonics* 32.6 (1994), pp. 431–440 (cit. on p. 49).
- [42]Leonid Brekhovskikh. *Waves in layered media*. Vol. 16. Elsevier, 2012 (cit. on p. 49).
- [43]Ramazan Demirli and Jafar Saniie. “Model-based estimation of ultrasonic echoes. Part I: Analysis and algorithms”. In: *IEEE transactions on ultrasonics, ferroelectrics, and frequency control* 48.3 (2001), pp. 787–802 (cit. on pp. 49, 51, 52).
- [44]Ramazan Demirli and Jafar Saniie. “Model-based estimation pursuit for sparse decomposition of ultrasonic echoes”. In: *IET signal processing* 6.4 (2012), pp. 313–325 (cit. on p. 49).
- [45]Stéphane G Mallat and Zhifeng Zhang. “Matching pursuits with time-frequency dictionaries”. In: *IEEE Transactions on signal processing* 41.12 (1993), pp. 3397–3415 (cit. on pp. 49, 52).

- [46]Joel A Tropp and Anna C Gilbert. “Signal recovery from random measurements via orthogonal matching pursuit”. In: *IEEE Transactions on information theory* 53.12 (2007), pp. 4655–4666 (cit. on pp. 49, 52, 57).
- [47]Etai Mor, Amnon Azoulay, and Mayer Aladjem. “A matching pursuit method for approximating overlapping ultrasonic echoes”. In: *IEEE transactions on ultrasonics, ferroelectrics, and frequency control* 57.9 (2010), pp. 1996–2004 (cit. on pp. 49, 51, 52, 57).
- [48]Ingrid Daubechies, Michel Defrise, and Christine De Mol. “An iterative thresholding algorithm for linear inverse problems with a sparsity constraint”. In: *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences* 57.11 (2004), pp. 1413–1457 (cit. on pp. 49, 52, 57).
- [49]Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep learning”. In: *nature* 521.7553 (2015), p. 436 (cit. on p. 50).
- [50]Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. “Wavenet: A generative model for raw audio”. In: *arXiv preprint arXiv:1609.03499* (2016) (cit. on pp. 50, 55).
- [51]Jean-Pierre Briot, Gaëtan Hadjeres, and François-David Pachet. “Deep learning techniques for music generation—a survey”. In: *arXiv preprint arXiv:1709.01620* (2017) (cit. on p. 50).
- [52]Hendrik Purwins, Bo Li, Tuomas Virtanen, Jan Schlüter, Shuo-Yiin Chang, and Tara Sainath. “Deep learning for audio signal processing”. In: *IEEE Journal of Selected Topics in Signal Processing* 13.2 (2019), pp. 206–219 (cit. on p. 50).
- [53]Yong Xu, Jun Du, Li-Rong Dai, and Chin-Hui Lee. “A regression approach to speech enhancement based on deep neural networks”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 23.1 (2014), pp. 7–19 (cit. on p. 50).
- [54]Santiago Pascual, Antonio Bonafonte, and Joan Serra. “SEGAN: Speech enhancement generative adversarial network”. In: *arXiv preprint arXiv:1703.09452* (2017) (cit. on p. 50).
- [55]Dario Rethage, Jordi Pons, and Xavier Serra. “A wavenet for speech denoising”. In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2018, pp. 5069–5073 (cit. on pp. 50, 55).
- [56]Yael Konforti, Alon Shpigler, Boaz Lerner, and Aharon Bar-Hillel. “Inference Graphs for CNN Interpretation”. In: *European Conference on Computer Vision*. Springer. 2020, pp. 69–84 (cit. on p. 50).
- [57]Karol Gregor and Yann LeCun. “Learning fast approximations of sparse coding”. In: *Proceedings of the 27th International Conference on International Conference on Machine Learning*. Omnipress. 2010, pp. 399–406 (cit. on p. 50).
- [58]Vishal Monga, Yuelong Li, and Yonina C Eldar. “Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing”. In: *arXiv preprint arXiv:1912.10557* (2019) (cit. on p. 50).
- [59]Zhipeng Li, Tong Wu, Wei Zhang, Xuyang Gao, Zhenqiu Yao, Yanjun Li, and Yibing Shi. “A Study on Determining Time-Of-Flight Difference of Overlapping Ultrasonic Signal: Wave-Transform Network”. In: *Sensors* 20.18 (2020), p. 5140 (cit. on p. 50).
- [60]Kjetil F Kaaresen. “Deconvolution of sparse spike trains by iterated window maximization”. In: *IEEE Transactions on Signal Processing* 45.5 (1997), pp. 1173–1183 (cit. on p. 51).
- [61]Michael S O’Brien, Anthony N Sinclair, and Stuart M Kramer. “Recovery of a sparse spike time series by  $L_{1/\text{norm}}$  deconvolution”. In: *IEEE Transactions on Signal Processing* 42.12 (1994), pp. 3353–3365 (cit. on p. 51).

- [62]Ramazan Demirli and Jafar Saniie. “A generic parametric model for ultrasonic signal analysis”. In: *2009 IEEE International Ultrasonics Symposium*. IEEE. 2009, pp. 1522–1525 (cit. on p. 52).
- [63]Florian Boßmann, Gerlind Plonka, Thomas Peter, Oliver Nemitz, and Till Schmitte. “Sparse deconvolution methods for ultrasonic NDT”. In: *Journal of Nondestructive Evaluation* 31.3 (2012), pp. 225–244 (cit. on p. 52).
- [64]Jonathan Long, Evan Shelhamer, and Trevor Darrell. “Fully convolutional networks for semantic segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3431–3440 (cit. on p. 53).
- [65]Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems*. 2012, pp. 1097–1105 (cit. on p. 54).
- [66]Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. “Dropout: a simple way to prevent neural networks from overfitting”. In: *The journal of machine learning research* 15.1 (2014), pp. 1929–1958 (cit. on p. 55).
- [67]Victor Lempitsky and Andrew Zisserman. “Learning to count objects in images”. In: *Advances in neural information processing systems* 23 (2010), pp. 1324–1332 (cit. on p. 56).
- [68]Yotam Itzhaky, Guy Farjon, Faina Khoroshevsky, Alon Shpigler, and Aharon Bar-Hillel. “Leaf counting: Multiple scale regression and detection using deep CNNs.” In: *BMVC*. 2018, p. 328 (cit. on p. 56).
- [69]Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014) (cit. on p. 56).
- [70]Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. “Pytorch: An imperative style, high-performance deep learning library”. In: *Advances in neural information processing systems*. 2019, pp. 8026–8037 (cit. on p. 56).
- [71]Zachary Pincus and JA Theriot. “Comparison of quantitative methods for cell-shape analysis”. In: *Journal of microscopy* 227.2 (2007), pp. 140–156 (cit. on p. 64).
- [72]Kinneret Keren, Zachary Pincus, Greg M Allen, Erin L Barnhart, Gerard Marriott, Alex Mogilner, and Julie A Theriot. “Mechanism of shape determination in motile cells”. In: *Nature* 453.7194 (2008), pp. 475–480 (cit. on p. 64).
- [73]Pei-Hsun Wu, Daniele M Gilkes, Jude M Phillip, Akshay Narkar, Thomas Wen-Tao Cheng, Jorge Marchand, Meng-Horng Lee, Rong Li, and Denis Wirtz. “Single-cell morphology encodes metastatic potential”. In: *Science advances* 6.4 (2020), eaaw6938 (cit. on p. 64).
- [74]Jonathan R Friedman, Laura L Lackner, Matthew West, Jared R DiBenedetto, Jodi Nunnari, and Gia K Voeltz. “ER tubules mark sites of mitochondrial division”. In: *Science* 334.6054 (2011), pp. 358–362 (cit. on p. 64).
- [75]Haoxi Wu, Pedro Carvalho, and Gia K Voeltz. “Here, there, and everywhere: The importance of ER membrane contact sites”. In: *Science* 361.6401 (2018), eaan5835 (cit. on pp. 64, 65).
- [76]Nadav Shai, Eden Yifrach, Carlo WT van Roermund, Nir Cohen, Chen Bibi, Lodewijk IJlst, Laetitia Cavellini, Julie Meurisse, Ramona Schuster, Lior Zada, et al. “Systematic mapping of contact sites reveals tethers and a function for the peroxisome-mitochondria contact”. In: *Nature communications* 9.1 (2018), p. 1761 (cit. on p. 65).



- [77]Luca Scorrano, Maria Antonietta De Matteis, Scott Emr, Francesca Giordano, György Hajnóczky, Benoît Kornmann, Laura L Lackner, Tim P Levine, Luca Pellegrini, Karin Reinisch, et al. “Coming together to define membrane contact sites”. In: *Nature communications* 10.1 (2019), p. 1287 (cit. on p. 65).
- [78]Sarah Cohen, Alex M Valm, and Jennifer Lippincott-Schwartz. “Interacting organelles”. In: *Current opinion in cell biology* 53 (2018), pp. 84–91 (cit. on p. 65).
- [79]Juan C Caicedo, Sam Cooper, Florian Heigwer, Scott Warchal, Peng Qiu, Csaba Molnar, Aliaksei S Vasilevich, Joseph D Barry, Harmanjit Singh Bansal, Oren Kraus, et al. “Data-analysis strategies for image-based cell profiling”. In: *Nature methods* 14.9 (2017), pp. 849–863 (cit. on p. 65).
- [80]Mark-Anthony Bray, Sigrun M Gustafsdottir, Mohammad H Rohban, Shantanu Singh, Vebjorn Ljosa, Katherine L Sokolnicki, Joshua A Bittker, Nicole E Bodycombe, Vlado Dančík, Thomas P Hasaka, et al. “A dataset of images and morphological profiles of 30 000 small-molecule treatments using the Cell Painting assay”. In: *Gigascience* 6.12 (2017), giw014 (cit. on pp. 65, 68, 70).
- [81]Assaf Zaritsky, Yun-Yu Tseng, M Angeles Rabadán, Shefali Krishna, Michael Overholtzer, Gaudenz Danuser, and Alan Hall. “Diverse roles of guanine nucleotide exchange factors in regulating collective cell migration”. In: *Journal of Cell Biology* 216.6 (2017), pp. 1543–1556 (cit. on p. 65).
- [82]Srinivas Niranj Chandrasekaran, Hugo Ceulemans, Justin D Boyd, and Anne E Carpenter. “Image-based profiling for drug discovery: due for a machine-learning upgrade?” In: *Nature Reviews Drug Discovery* 20.2 (2021), pp. 145–159 (cit. on p. 65).
- [83]Gregory P Way, Ted Natoli, Adeniyi Adeboye, Lev Litichevskiy, Andrew Yang, Xiaodong Lu, Juan C Caicedo, Beth A Cimini, Kyle Karhohs, David J Logan, et al. “Morphology and gene expression profiling provide complementary information for mapping cell state”. In: *Cell systems* 13.11 (2022), pp. 911–923 (cit. on pp. 65, 68, 70, 71, 78).
- [84]Ben T Grys, Dara S Lo, Nil Sahin, Oren Z Kraus, Quaid Morris, Charles Boone, and Brenda J Andrews. “Machine learning and computer vision approaches for phenotypic profiling”. In: *Journal of Cell Biology* 216.1 (2017), pp. 65–71 (cit. on p. 65).
- [85]Mark-Anthony Bray, Shantanu Singh, Han Han, Chadwick T Davis, Blake Borgeson, Cathy Hartland, Maria Kost-Alimova, Sigrun M Gustafsdottir, Christopher C Gibson, and Anne E Carpenter. “Cell Painting, a high-content image-based assay for morphological profiling using multiplexed fluorescent dyes”. In: *Nature protocols* 11.9 (2016), pp. 1757–1774 (cit. on pp. 65, 77).
- [86]David R Stirling, Madison J Swain-Bowden, Alice M Lucas, Anne E Carpenter, Beth A Cimini, and Allen Goodman. “CellProfiler 4: improvements in speed, utility and usability”. In: *BMC bioinformatics* 22 (2021), pp. 1–11 (cit. on pp. 65, 66, 77).
- [87]Mechanism Matters. “Mechanism matters”. In: *Nat. Med* 16.4 (2010), pp. 347–347 (cit. on pp. 65, 71).
- [88]Marzieh Haghighi, Juan C Caicedo, Beth A Cimini, Anne E Carpenter, and Shantanu Singh. “High-dimensional gene expression and morphology profiles of cells across 28,000 genetic and chemical perturbations”. In: *Nature methods* 19.12 (2022), pp. 1550–1557 (cit. on pp. 65, 66, 68, 70, 71, 78, 79).
- [89]Nikita Moshkov, Tim Becker, Kevin Yang, Peter Horvath, Vlado Dancik, Bridget K Wagner, Paul A Clemons, Shantanu Singh, Anne E Carpenter, and Juan C Caicedo. “Predicting compound activity from phenotypic profiles and chemical structures”. In: *Nature Communications* 14.1 (2023), p. 1967 (cit. on pp. 65, 71).



- [90]Shinsuke Ohnuki, Itsuki Ogawa, Kaori Itto-Nakama, Fachuang Lu, Ashish Ranjan, Mehdi Kabbage, Abraham Abera Gebre, Masao Yamashita, Sheena C Li, Yoko Yashiroda, et al. “High-throughput platform for yeast morphological profiling predicts the targets of bioactive compounds”. In: *npj Systems Biology and Applications* 8.1 (2022), p. 3 (cit. on p. 65).
- [91]Juan C Caicedo, John Arevalo, Federica Piccioni, Mark-Anthony Bray, Cathy L Hartland, Xiaoyun Wu, Angela N Brooks, Alice H Berger, Jesse S Boehm, Anne E Carpenter, et al. “Cell Painting predicts impact of lung cancer variants”. In: *Molecular biology of the cell* 33.6 (2022), ar49 (cit. on pp. 65, 68, 70, 78).
- [92]Shantanu Singh, Xiaoyun Wu, Vebjorn Ljosa, Mark-Anthony Bray, Federica Piccioni, David E Root, John G Doench, Jesse S Boehm, and Anne E Carpenter. “Morphological profiles of RNAi-induced gene knockdown are highly reproducible but dominated by seed effects”. In: *PloS one* 10.7 (2015), e0131370 (cit. on p. 65).
- [93]Matheus P Viana, Jianxu Chen, Theo A Knijnenburg, Ritvik Vasan, Calysta Yan, Joy E Arakaki, Matte Bailey, Ben Berry, Antoine Borensztein, Eva M Brown, et al. “Integrated intracellular organization and its variations in human iPS cells”. In: *Nature* 613.7943 (2023), pp. 345–354 (cit. on p. 65).
- [94]Srikanth Thudumu, Philip Branch, Jiong Jin, and Jugdutt Singh. “A comprehensive survey of anomaly detection techniques for high dimensional big data”. In: *Journal of Big Data* 7 (2020), pp. 1–30 (cit. on p. 65).
- [95]Hiroyuki Kobayashi, Keith C Cheveralls, Manuel D Leonetti, and Loic A Royer. “Self-supervised deep learning encodes high-resolution features of protein subcellular localization”. In: *Nature methods* 19.8 (2022), pp. 995–1003 (cit. on pp. 65, 76).
- [96]Nikita Moshkov, Michael Bornholdt, Santiago Benoit, Matthew Smith, Claire McQuin, Allen Goodman, Rebecca A Senft, Yu Han, Mehrtash Babadi, Peter Horvath, et al. “Learning representations for image-based profiling of perturbations”. In: *Nature Communications* 15.1 (2024), p. 1594 (cit. on pp. 65, 76).
- [97]Anastasia Razdaibiedina, Alexander Brechalov, Helena Friesen, Mojca Mattiazzi Usaj, Myra Paz David Masinas, Harsha Garadi Suresh, Kyle Wang, Charles Boone, Jimmy Ba, and Brenda Andrews. “PIFiA: self-supervised approach for protein functional annotation from single-cell imaging data”. In: *Molecular Systems Biology* (2024), pp. 1–28 (cit. on pp. 65, 76).
- [98]Lena Molitor, Sagy Krispin, Welmoed Van-Zuiden, Yehud M Danino, Noam Rudberg, Chen Bar, Emmanuel Amzallag, Jazz Lubliner, Aviad Siany, Chen Eitan, et al. “Organellomics: AI-driven deep organellar phenotyping of human neurons”. In: *bioRxiv* (2024), pp. 2024–01 (cit. on pp. 65, 76).
- [99]Alex X Lu, Oren Z Kraus, Sam Cooper, and Alan M Moses. “Learning unsupervised feature representations for single cell microscopy images with paired cell inpainting”. In: *PLoS computational biology* 15.9 (2019), e1007348 (cit. on pp. 65, 76).
- [100]Robert van Dijk, John Arevalo, Mehrtash Babadi, Anne E Carpenter, and Shantanu Singh. “Capturing cell heterogeneity in representations of cell populations for image-based profiling using contrastive learning”. In: *bioRxiv* (2023), pp. 2023–11 (cit. on pp. 65, 76).
- [101]Oren Kraus, Kian Kenyon-Dean, Saber Saberian, Maryam Fallah, Peter McLean, Jess Leung, Vasudev Sharma, Ayla Khan, Jia Balakrishnan, Safiye Celik, et al. “Masked autoencoders are scalable learners of cellular morphology”. In: *arXiv preprint arXiv:2309.16064* (2023) (cit. on p. 65).
- [102]Varun Chandola, Arindam Banerjee, and Vipin Kumar. “Anomaly detection: A survey”. In: *ACM computing surveys (CSUR)* 41.3 (2009), pp. 1–58 (cit. on p. 65).

- [103]Tharindu Fernando, Harshala Gammulle, Simon Denman, Sridha Sridharan, and Clinton Fookes. “Deep learning for medical anomaly detection—a survey”. In: *ACM Computing Surveys (CSUR)* 54.7 (2021), pp. 1–37 (cit. on p. 65).
- [104]Changhee Han, Leonardo Rundo, Kohei Murao, Tomoyuki Noguchi, Yuki Shimahara, Zoltán Ádám Milacski, Saori Koshino, Evis Sala, Hideki Nakayama, and Shin’ichi Satoh. “MADGAN: Unsupervised medical anomaly detection GAN using multiple adjacent brain MRI slice reconstruction”. In: *BMC bioinformatics* 22 (2021), pp. 1–20 (cit. on p. 65).
- [105]Krishnan Naidoo and Vukosi Marivate. “Unsupervised anomaly detection of healthcare providers using generative adversarial networks”. In: *Responsible Design, Implementation and Use of Information and Communication Technology: 19th IFIP WG 6.11 Conference on e-Business, e-Services, and e-Society, I3E 2020, Skukuza, South Africa, April 6–8, 2020, Proceedings, Part I* 19. Springer. 2020, pp. 419–430 (cit. on p. 65).
- [106]Jorge Meira, Rui Andrade, Isabel Praça, João Carneiro, Verónica Bolón-Canedo, Amparo Alonso-Betanzos, and Goreti Marreiros. “Performance evaluation of unsupervised techniques in cyber-attack anomaly detection”. In: *Journal of Ambient Intelligence and Humanized Computing* 11.11 (2020), pp. 4477–4489 (cit. on p. 65).
- [107]Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. “LOF: identifying density-based local outliers”. In: *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*. 2000, pp. 93–104 (cit. on p. 65).
- [108]Ville Hautamaki, Ismo Karkkainen, and Pasi Franti. “Outlier detection using k-nearest neighbour graph”. In: *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004*. Vol. 3. IEEE. 2004, pp. 430–433 (cit. on p. 65).
- [109]Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. “Isolation forest”. In: *2008 eighth ieee international conference on data mining*. IEEE. 2008, pp. 413–422 (cit. on p. 65).
- [110]Guansong Pang, Chunhua Shen, Longbing Cao, and Anton Van Den Hengel. “Deep learning for anomaly detection: A review”. In: *ACM computing surveys (CSUR)* 54.2 (2021), pp. 1–38 (cit. on p. 65).
- [111]Jingkang Yang, Kaiyang Zhou, Yixuan Li, and Ziwei Liu. “Generalized out-of-distribution detection: A survey”. In: *arXiv preprint arXiv:2110.11334* (2021) (cit. on p. 66).
- [112]Lukas Ruff, Jacob R Kauffmann, Robert A Vandermeulen, Grégoire Montavon, Wojciech Samek, Marius Kloft, Thomas G Dietterich, and Klaus-Robert Müller. “A unifying review of deep and shallow anomaly detection”. In: *Proceedings of the IEEE* 109.5 (2021), pp. 756–795 (cit. on p. 66).
- [113]Mohammad Hossein Rohban, Shantanu Singh, Xiaoyun Wu, Julia B Berthet, Mark-Anthony Bray, Yashaswi Shrestha, Xaralabos Varelas, Jesse S Boehm, and Anne E Carpenter. “Systematic morphological profiling of human gene and allele function via Cell Painting”. In: *Elife* 6 (2017), e24060 (cit. on pp. 68, 70).
- [114]Nalini Schaduangrat, Samuel Lampa, Saw Simeon, Matthew Paul Gleeson, Ola Spjuth, and Chanin Nantasenamat. “Towards reproducible computational drug discovery”. In: *Journal of cheminformatics* 12 (2020), pp. 1–30 (cit. on p. 68).
- [115]Erin Weisbart, Ankur Kumar, John Arevalo, Anne E Carpenter, Beth A Cimini, and Shantanu Singh. “Cell Painting Gallery: an open resource for image-based profiling”. In: *arXiv preprint arXiv:2402.02203* (2024) (cit. on pp. 70, 77).
- [116]Daniel R Wong, David J Logan, Santosh Hariharan, Robert Stanton, Djork-Arné Clevert, and Andrew Kiruluta. “Deep representation learning determines drug mechanism of action from cell painting images”. In: *Digital Discovery* 2.5 (2023), pp. 1354–1367 (cit. on p. 71).

- [117] Scott M Lundberg and Su-In Lee. “A unified approach to interpreting model predictions”. In: *Advances in neural information processing systems* 30 (2017) (cit. on pp. 73, 79).
- [118] Liat Antwarg, Ronnie Mindlin Miller, Bracha Shapira, and Lior Rokach. “Explaining anomalies detected by autoencoders using Shapley Additive Explanations”. In: *Expert systems with applications* 186 (2021), p. 115736 (cit. on pp. 73, 79).
- [119] Meraj Ramezani, Julia Bauman, Avtar Singh, Erin Weisbart, John Yong, Maria Lozada, Gregory P Way, Sanam L Kavari, Celeste Diaz, Marzieh Haghighi, et al. “A genome-wide atlas of human cell morphology”. In: *bioRxiv* (2023) (cit. on pp. 73, 78).
- [120] Gregory P Way, Maria Kost-Alimova, Tsukasa Shibue, William F Harrington, Stanley Gill, Federica Piccioni, Tim Becker, Hamdah Shafqat-Abbasi, William C Hahn, Anne E Carpenter, et al. “Predicting cell health phenotypes using image-based morphology profiling”. In: *Molecular biology of the cell* 32.9 (2021), pp. 995–1005 (cit. on p. 75).
- [121] John Arevalo, Ellen Su, Robert van Dijk, Anne E Carpenter, and Shantanu Singh. “Evaluating batch correction methods for image-based cell profiling”. In: *bioRxiv* (2023) (cit. on p. 75).
- [122] Paula A Marin Zapata, Oscar Méndez-Lucio, Tuan Le, Carsten Jörn Beese, Jörg Wichard, David Rouquié, and Djork-Arné Clevert. “Cell morphology-guided de novo hit design by conditioning GANs on phenotypic image features”. In: *Digital Discovery* 2.1 (2023), pp. 91–102 (cit. on p. 76).
- [123] Christopher J Soelistyo, Giulia Vallardi, Guillaume Charras, and Alan R Lowe. “Learning biophysical determinants of cell fate with deep neural networks”. In: *Nature Machine Intelligence* 4.7 (2022), pp. 636–644 (cit. on p. 76).
- [124] Dong Gong, Lingqiao Liu, Vuong Le, Budhaditya Saha, Moussa Reda Mansour, Svetha Venkatesh, and Anton van den Hengel. “Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 1705–1714 (cit. on p. 76).
- [125] Aravind Subramanian, Rajiv Narayan, Steven M Corsello, David D Peck, Ted E Natoli, Xiaodong Lu, Joshua Gould, John F Davis, Andrew A Tubelli, Jacob K Asiedu, et al. “A next generation connectivity map: L1000 platform and the first 1,000,000 profiles”. In: *Cell* 171.6 (2017), pp. 1437–1452 (cit. on p. 77).
- [126] Erik Serrano, Srinivas Niranj Chandrasekaran, Dave Bunten, Kenneth I Brewer, Jenna Tomkinson, Roshan Kern, Michael Bornholdt, Stephen Fleming, Ruifan Pei, John Arevalo, et al. “Reproducible image-based profiling with Pycytominer”. In: *ArXiv* (2023) (cit. on p. 78).

## תקציר

עבודת גמר זו חוקרת יישום למידה עמוקה עבור שלושה תחומים מדעיים ואופני רכישת תמונות שונים: מתן פרשנות עבור אלגוריתמי ראייה ממוחשבת בתמונות טבעיות (פרק 1), חיזוי עומק גופים באולטרסאונד (פרק 2), ומציאת מועמדות לתרופות במיקרוסקופיה תאית (פרק 3). הפרק הראשון כולל פיתוח של גרפי השערה סטטיסטיים המבוססים על מודלים הסתברותיים לפרשנות של רשתות נוירונים עמוקות עבור שיטות ראייה ממוחשבת. הגרפים מספקים שכבה של שקיפות והבנה של תהליכי קבלת החלטות של הרשתות. הפרק השני מתמקד בגילוי והפרדה של הדים אולטרסוניים חופפים באמצעות רשתות עמוקות, במטרה לשפר את חיזוי העומק עבור יישומי בדיקות לא הורסות. הפרק השלישי מציג שיטת גילוי חריגות עבור ניסויי ביולוגיה תאית מבוססי תמונה. השיטה מסייעת לגלות תבניות שהשתנו במבנים תאיים כתוצאה ממתן תרופות באמצעות רשתות עמוקות. למרות השוני ביניהם, כל הפרקים בעבודת גמר זו משתמשים בטכניקות חדשניות בלמידה עמוקה.

פרשנות של רשתות נוירונים עמוקות ויישומים עבור אולטרסאונד וביולוגיה תאית

מחקר לשם מילוי חלקי של הדרישות לקבלת תואר "דוקטור לפילוסופיה"

מאת

אלון שפיגלר

הוגש לסינאט אוניברסיטת בן גוריון בנגב

אישור המנחה פרופ' אסף זריצקי  
אישור המנחה פרופ' הלל בר-גרא

אישור דיקן בית הספר ללימודי מחקר מתקדמים ע"ש קרייטמן

07.05.2024

כ"ט בניסן תשפ"ד

באר שבע

**פרשנות של רשתות נוירונים עמוקות ויישומים עבור אולטרסאונד וביולוגיה תאית**

**מחקר לשם מילוי חלקי של הדרישות לקבלת תואר "דוקטור לפילוסופיה"**

**מאת**

**אלון שפיגלר**

**הוגש לסינאט אוניברסיטת בן גוריון בנגב**

**07.05.2024**

**כ"ט בניסן תשפ"ד**

**באר שבע**