Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

# A parameter-efficient deep ensemble for automated flow cytometry-based diagnosis

**Tomasz Gniazdowski**

gniazdowski.tomek@gmail.com

Wrocław University of Science and Technology

**Urszula Markowska-Kaczmar**

Wrocław University of Science and Technology

**Article**

**Keywords:**

**Posted Date:** June 30th, 2026

**DOI:** https://doi.org/10.21203/rs.3.rs-10126876/v1

**License:** This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

**Additional Declarations:** Competing interest reported. Urszula Markowska-Kaczmar was an editor at Scientific Reports for a year, until January of 2026.

**Version of Record:** A version of this preprint was published at Scientific Reports on October 12th, 2026. See the published version at https://doi.org/10.1038/s41598-026-74128-9.

# A parameter-efficient deep ensemble for automated flow cytometry-based diagnosis

**Tomasz Gniazdowski**[1,*] **and Urszula Markowska-Kaczmar**[1]

[1]Artificial Intelligence Department, Wroclaw University of Science and Technology, Wroclaw, Wyb. Wyspianskiego 27 50-370, Poland
[*]gniazdowski.tomek@gmail.com

## ABSTRACT

Flow cytometry is a cornerstone diagnostic technique, but manual analysis is laborious, subjective, and prone to inter-operator variability. We propose a parameter-efficient deep ensemble model, *CytoDNNEnsemble*, for automated analysis of high-dimensional flow cytometry data. The architecture consists of two smaller, independent subnetworks that process blood samples from complementary global and local perspectives. This design reduces the reliance on expert gating, captures complex cellular patterns, and improves diagnostic consistency. Extensive experiments on relatively large real-world datasets demonstrate that the proposed model outperforms state-of-the-art approaches in terms of accuracy and robustness while keeping the parameter count low, highlighting its potential for clinical deployment.

## Introduction

Flow cytometry is a laboratory technique that is used to analyze the physical and chemical characteristics of individual cells suspended in a fluid. It is a cornerstone of hematology, oncology, immunology, and transplant medicine, and is widely used to diagnose conditions such as leukemia, lymphoma, HIV infection, immune deficiencies, and autoimmune disorders.

In a typical workflow, blood, bone marrow, or tissue samples are stained with fluorescently labeled antibodies that bind to specific cellular markers, enabling differentiation of cell types and characterization of their activation or maturation states. The stained sample is then processed by a flow cytometer, which measures light scatter and fluorescence intensities for thousands to millions of individual cells, representing each cell as a high-dimensional vector.

Subsequently, clinicians or technicians perform manual gating by drawing polygons on two-dimensional plots to identify distinct cell populations. This step is highly dependent on expert judgment and is guided by visual heuristics and prior experience. Based on the distribution of cell populations, abnormal marker expression, or absence of expected populations, the clinician establishes a diagnosis or recommends further testing.

However, several factors make this diagnostic process challenging. First, modern cytometers routinely measure 20–50 or more markers per cell, making manual analysis labor-intensive and error-prone. Second, manual gating is inherently subjective and not standardized, leading to substantial inter-expert variability and making diagnostic accuracy strongly dependent on individual expertise. Third, the time-consuming nature of manual analysis can delay diagnosis in urgent clinical settings.

These limitations underscore the need for automated, reliable, and interpretable AI tools for flow cytometry-based diagnosis. Such tools can reduce workload, improve consistency, and uncover non-obvious marker patterns and cell populations that may be missed in manual inspection. Deep learning models–in particular ensemble-based approaches are well suited to capture complex interactions in high-dimensional biomedical data and to detect rare cell phenotypes that are difficult for humans to identify[1–5].

Nevertheless, flow cytometry presents unique challenges for deep learning, including sparsity, batch effects, and complex cell population distributions[6,7]. Deep ensembles are widely recognized for improving predictive accuracy and uncertainty estimation[8], but their high computational and memory requirements hinder integration into routine diagnostic pipelines. Moreover, most ensemble methods have been developed for domains such as computer vision[9–11] or natural language processing[12–14] and are not optimized for the structure and variability of flow cytometry data. As a result, there remains a notable gap in the development of parameter-efficient deep ensemble models that can balance predictive performance, uncertainty calibration, and deployment efficiency in cytometry-based diagnostic support systems.

In this work, we address this gap by introducing a parameter-efficient deep ensemble model tailored to flow cytometry-based diagnosis. Our main contributions are as follows:

- We propose a novel parameter-efficient deep ensemble architecture for flow cytometry-based diagnosis that takes advantage of complementary global and local representations of blood samples.
- We design a scalable ensemble strategy that improves robustness and uncertainty estimation without incurring a prohibitive computational or memory burden, facilitating clinical deployment.
- We conduct a comprehensive evaluation on relatively large real-world datasets, demonstrating superior predic-

tive performance and reliability compared to state-of-the-art methods.

## Related work

Recent advances in flow and imaging cytometry have enabled high-throughput measurement of cellular features, creating a demand for robust computational methods capable of handling the resulting high-dimensional data. Classical approaches, such as manual clustering and gating-based techniques, often suffer from limited scalability, reproducibility, and robustness in real-world clinical settings[15]. Early machine learning methods, including decision trees[16], random forests[17], and support vector machines (SVMs)[18], were applied to automate cytometry analysis as supervised learning models that rely on annotated datasets. However, obtaining large, reliably annotated cytometry datasets is challenging, time-consuming, and expensive.

To address the scarcity of labels, many studies have adopted unsupervised learning approaches, particularly clustering techniques that do not require annotations. Examples include K-means clustering[19], FlowGrid, which builds on DBSCAN-based clustering[20], and Gaussian mixture modeling as implemented in PhenoGMM[21]. Dimensionality reduction methods are also widely used to visualize and interpret high-dimensional cytometry data, including Principal Component Analysis (PCA), Self-Organizing Maps as in FlowSOM[22], and nonlinear techniques such as t-SNE and UMAP[23]. A recent survey summarizes classical and modern machine learning approaches for flow cytometry analysis[24]. In parallel, masked autoencoders have been explored as a special class of autoencoders tailored to cytometry, aiming to learn task-agnostic feature representations that can support downstream tasks such as classification, detection, or segmentation[25].

With the development of deep learning, recent research has focused on extracting informative representations (embeddings) using deep models[26,27]. These embeddings can be further used for classification or clustering, making them attractive building blocks for intelligent diagnostic systems. One common approach is the use of autoencoders[28], which consist of an encoder and a decoder trained to reconstruct the input; the bottleneck layer provides a compressed representation of the data that can serve as a compact feature space for downstream tasks.

Several works have applied autoencoders directly to flow cytometry. For example, Zhang et al.[29] use learned embeddings for clustering and analyze which patterns the model exploits to separate cell populations, contributing to interpretability. Kim et al.[30] demonstrated the effectiveness of a masked autoencoder for learning relationships between protein markers in immune cells based solely on protein expression data, showing that self-supervised reconstruction can capture biologically meaningful structure.

A novel computational framework based on variational autoencoders (VAEs) was introduced in[31] to integrate unpaired biomedical data streams in flow cytometry. The VAE model[32] is a generative deep learning approach, also used in the FlowVI model[33], which achieves high accuracy in recovering missing protein markers and supports unified analysis across heterogeneous cytometry panels. Such generative models are particularly relevant for clinical decision support systems, where harmonizing data from multiple instruments and panels is essential.

As discussed earlier, building large annotated datasets for supervised learning in cytometry is very difficult and requires laborious expert work. This has motivated the development of self-supervised methods that can leverage large unlabeled datasets together with a limited annotated subset. Self-supervised learning is commonly used in pre-training pipelines and may include teacher–student frameworks or other unsupervised objectives to construct meaningful embedding spaces from diverse datasets. Well-known self-supervised techniques such as BYOL[34], SimCLR[35], and DINOv2[36] exemplify this approach and have been successfully applied to single-cell data[37,38], improving downstream performance and data efficiency.

Recent studies have also shifted part of the focus from traditional flow cytometry—which measures multiple physical and chemical characteristics of cells (e.g., size, granularity, fluorescence from labeled antibodies) as they pass single-file through a laser beam—toward imaging flow cytometry. The latter captures high-resolution images of individual cells as they flow through the instrument, producing bright-field, fluorescence, and phase-contrast images. It supports both quantitative analysis of marker expression (as in conventional flow cytometry) and qualitative or spatial analyses such as subcellular localization, co-localization, morphology, or cell–cell interactions. Examples of such work include automated pipelines for imaging cytometry and predictive modeling based on cell images[37,39,40].

The backbone models used in cytometry, especially in imaging flow cytometry, include convolutional neural networks such as ResNet[41] and EfficientNet[42], which are widely adopted for image-based cell profiling. Vision Transformers (ViTs)[43], often trained using masked autoencoding objectives, have also been explored in this context[1]. ResNet-based feature extractors have been used, for example, for disease and cell-cycle modeling[44]. For non-image cytometry data, common backbones include multilayer perceptrons (MLPs)[45] and architectures adapted to tabular data such as TabNet[46], Transformers for automated identification of cell populations[47], and VAE encoders as probabilistic representation learners[24]. Recent review papers provide broader overviews of classical machine learning methods applied to flow cytometry[48,49].

As in other clinical domains, researchers in flow cytometry have increasingly focused on interpretable and explainable AI solutions[50,51]. Interpretability refers to models whose internal mechanisms can be directly understood by clinicians, while explainable AI encompasses post hoc techniques that explain the behavior of complex, black-box models such as deep neural networks. These approaches aim to increase

trust in AI-assisted diagnostic systems by providing human-understandable rationales for predictions, which is particularly important for high-stakes medical decisions.

Deep learning in flow cytometry has also progressed to the point of clinical deployment. Zuromski et al.[52] describe a real-time system and compare its clinical accuracy with the original validation statistics, showing that deep learning-based cytometry pipelines can satisfy robustness and reliability requirements in practice. Beyond single-model architectures, ensemble deep learning has emerged as an effective strategy for improving predictive performance and robustness in medical applications[53–55], often providing better uncertainty estimates than individual networks[56,57]. However, existing ensembles typically rely on large backbones or multiple fully independent models, which limits their suitability for real-time analysis on standard clinical hardware. In contrast, our work introduces a parameter-efficient deep ensemble tailored to flow cytometry-based diagnosis, explicitly designed to balance predictive performance, robustness, and computational efficiency.

## Material and Methods

In this section, we describe the cytometry datasets used in our study and the proposed deep ensemble model for flow cytometry-based diagnosis.

### Cytometry data

In this work, a sample refers to a single datapoint from an individual, represented as a matrix of size $[num_cells, num_markers]$, where each row corresponds to a cell and each column to a marker. This representation serves as the input for multi-cell cytometry-based classification, enabling the model to learn patterns across both cells and markers. The number of samples of a given class and the number of samples in the training and test datasets are presented in the Table 1.

#### *AML*

The AML dataset consists of 359 flow cytometry samples collected from individuals with acute myeloid leukemia (AML) and non-AML controls. Each sample contains measurements of multiple surface and intracellular markers on peripheral blood or bone marrow cells. In our experiments, the dataset is randomly split into training (including validation) and test subsets with an 80%:20% ratio, ensuring that samples from the same individual appear in only one subset.

#### *HEUvsUE*

The HEUvsUE dataset comprises 308 samples from African infants. Positive samples correspond to HIV-exposed but uninfected children (HEU), while negative samples are from HIV-unexposed infants (UE). Each sample is represented by multi-parameter flow cytometry measurements of immune cell populations. For this benchmark, 80% of the samples are used for training and validation, and 20% are reserved for testing, following the protocol in previous work when applicable.

#### *HVTN*

The HVTN (HIV Vaccine Trials Network) dataset contains 96 samples obtained from T cells collected after HIV vaccination and stimulated with Gag or Env antigens. Following prior studies[15,58,59], we use 50% of the samples for training and 50% for testing. The split is performed at the sample level so that no individual contributes samples to both subsets.

#### *NK-Cell*

The NK-Cell dataset, originally used to evaluate the CellCNN model[60], consists of 20 expert-annotated NK-cell samples. Each sample is labeled as cytomegalovirus (CMV) seropositive or seronegative based on clinical assessment. We follow the train/test protocol described in the original CellCNN publication[60], using the same sample assignments where applicable to ensure comparability of results.

### CytoDNNEnsemble (CDE)

Before introducing the CytoDNNEnsemble, we briefly summarize the cytometry data representation used as input to the model. Data corresponding to a single cell form one row of a matrix, while each column refers to a specific marker; this row can be viewed as a feature vector describing one cell. A single sample contains multiple cells arranged in consecutive rows, resulting in a two-dimensional matrix of size $[num_cells, num_markers]$. The complete training dataset consists of multiple samples, which together form a three-dimensional tensor, as illustrated in Figure 1. Importantly, the proposed method is invariant to the ordering of rows: permuting the cells within a sample does not affect the model prediction.

The CytoDNNEnsemble (referred to as CDE) is presented in Figure 1. It consists of two independent models: a 1D Convolutional Neural Network and a Latent Space Extractor, hereafter referred to as CNN1DE and LSE, respectively. They both end with a classification layer. Their outputs are concatenated and fed to the final classifier that produces the output based on the combined predictions of both models.

From a machine learning perspective, cytometry-based diagnosis is formulated as a sample-level classification task. In all four datasets considered in this work, we address a binary classification problem, where the model is trained to assign each sample (i.e., a multi-cell matrix of marker values) to one of two clinically relevant classes. In principle, the same architecture can be extended to multi-class classification by adjusting only the size of the output layer.

The CytoDNNEnsemble (CDE) model is presented in Figure 1. It consists of two independent subnetworks: a one-dimensional Convolutional Neural Network Extractor (CNN1DE) and a Latent Space Extractor (LSE). Each subnetwork ends with its own classification head. Their outputs are concatenated and fed to a final classifier, which produces the sample-level prediction based on the combined information from both branches.

The main idea behind the proposed solution is that the two subnetworks analyze each sample from complementary per-

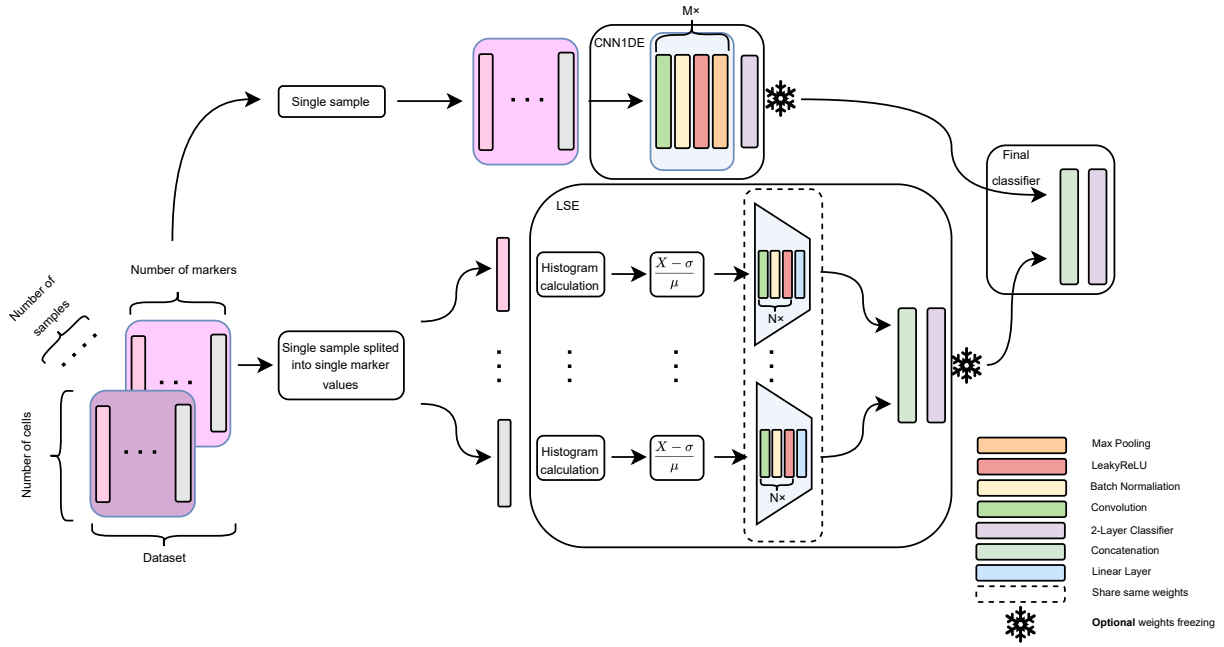



**Figure 1.** Overview of the CytoDNNEnsemble (CDE) model. Each sample with dimensionality $[num_cells, num_markers]$ is processed independently from two complementary perspectives. In the upper branch, the 1D Convolutional Neural Network Extractor (CNN1DE) processes the sample row by row using marker values of individual cells. In the lower branch, the Latent Space Extractor (LSE) separates marker-wise values (columns), computes and normalizes their histograms, and obtains latent representations using a pretrained convolutional encoder; these representations are concatenated and used for classification. The predictions of CNN1DE and LSE are concatenated and passed to a final classifier, which produces the final decision. CNN1DE and LSE are trained independently; during the final training stage, the weights of these subnetworks can either be frozen or the entire pipeline can be trained end-to-end. This design enables the ensemble to capture fine-grained cell-level information (CNN1DE) and global marker-level trends (LSE) within a parameter-efficient architecture.

spectives. A blood sample, represented as a matrix of size $[num_cells, num_markers]$ with $num_cells \gg num_markers$, is processed in two ways. The CNN1DE branch applies convolutional filters across marker values returned by individual cells, effectively analyzing the sample row by row and focusing on fine-grained, cell-level patterns. In contrast, the LSE branch aggregates marker values across all cells in a sample, derives marker-wise histograms, and uses a convolutional encoder to obtain latent representations; in this case, the sample is analyzed column by column, emphasizing global, marker-level trends.

By construction, CNN1DE captures local, cell-level information, while LSE captures global, marker-level characteristics. Combining these two complementary views allows the final classifier to operate on a richer and more comprehensive representation of the input, improving robustness and diagnostic performance without a prohibitive increase in the number of parameters. In the following, we describe both subnetworks and the final ensemble in more detail.

### *Latent Space Extractor (LSE)*

The Latent Space Extractor (LSE) branch is based on a convolutional encoder that is initially trained as part of a convolutional autoencoder. After this pretraining phase, only the encoder component is retained and used as the feature extractor in the LSE model. The autoencoder architecture comprises the following elements:

- **Convolutional encoder:** The encoder consists of two, three, or four convolutional blocks, corresponding to *small*, *medium*, and *large* variants of the model, respectively. Each block includes a 1D convolutional layer, a batch normalization layer, and a LeakyReLU activation function. The LeakyReLU activation[61] is employed to preserve negative values present in the input data, which would otherwise be suppressed by the ReLU activation. After the convolutional blocks, the encoder applies a flatten operation followed by a linear layer to obtain the desired dimensionality of the latent representation.

- **Convolutional decoder:** The decoder comprises a linear layer followed by an unflatten operation to restore the required data dimensionality. It then applies one, two, or three convolutional blocks (for small, medium, and large variants, respectively). Each decoder block consists of a 1D transposed convolutional layer, a batch normalization layer, and a LeakyReLU activation. All decoder variants additionally include a final 1D transposed convolutional layer as the last layer of the model.

By construction, the dimensionality of the input and output of the autoencoder is identical. The exact specification of the architecture, including the number of channels in each layer, kernel sizes, padding, and stride values, is provided in Appendix D.

During autoencoder training, each dataset sample with overall shape $[num_samples, num_cells, num_markers]$ is preprocessed as follows. First, each sample is independently

reshaped (Algorithm 1, line 1) to obtain a tensor of size $[num_samples \times num_markers, num_cells]$. This operation is implemented using the *permute* and *reshape* functions in PyTorch[62]. Next, histograms are computed along the last dimension (Algorithm 1, line 2), yielding a dataset of dimensionality $[num_samples \times num_markers, num_bins]$. In our experiments, a bin count of 100 was found to preserve the most important characteristics of the data while providing a compact representation, given that $num_cells \gg 100$.

Subsequently, histogram values are normalized (Algorithm 1, line 3). Let $\mathbf{X}_{\text{train}}$ denote the training data with shape $[num_samples \times num_markers, num_bins]$. The mean and standard deviation are computed as

$$\mu = \text{mean}(\mathbf{X}_{\text{train}}, \text{dim} = 0),$$

$$\sigma = \text{std_deviation}(\mathbf{X}_{\text{train}}, \text{dim} = 0).$$

Thus, $\mu$ and $\sigma$ are vectors of length *num_bins*. Any datapoint $\mathbf{X}$ is then normalized according to

$$\mathbf{X}' = \frac{\mathbf{X} - \mu}{\sigma}.$$

After this operation, each column $j$ has mean 0 and variance 1 across the training data, which improves the stability of model training.

The convolutional autoencoder is trained on normalized histograms using the loss of the mean squared error (MSE) (Algorithm 1, line 4), with the objective of reconstructing the input. This procedure yields a compressed hidden representation in the encoder that retains the most informative features of the original data. After training, the encoder is connected in series to a two-layer classifier (linear layer, LeakyReLU activation, linear layer, sigmoid activation), forming the LSE model; the decoder is discarded. A schematic illustration of the autoencoder and encoder is presented in Figure 2.

The complete LSE model, consisting of the pretrained convolutional encoder and the classifier, is then trained end-to-end on the binary classification task (Algorithm 1, lines 5–6) using the cross-entropy loss. For a given sample with shape $[num_markers, num_bins]$, the encoder produces latent representations of size $[num_markers, encoder_latent_dim]$. These representations are concatenated along the last dimension, resulting in a vector of length $num_markers \times encoder_latent_dim$, which is then processed by the classifier. While the scheme is illustrated for a single sample, all computations are implemented in a batched manner.

In the LSE branch, the labels *small*, *medium*, and *large* refer to the number of convolutional blocks rather than the total number of parameters. Because the latent dimensionality (32 or 64) is selected experimentally, it is possible that a small model with latent dimension 64 has more parameters than a medium model with latent dimension 32, mainly due to the contribution of linear layers. Specifically, a linear layer adds $linear_input_dim \times linear_output_dim$ parameters, while a 1D convolutional layer adds $kernel_length \times num_kernels$ parameters.

**Algorithm 1** LSE model training pipeline

**Require:** dataset // $x = [S, C, M]$, $y = [1]$
**Ensure:** trained LSE model // *Dimensions in comments: S – number of samples, C – number of cells, M – number of markers, H – number of histogram bins.*

1: $datasetAE \leftarrow flatten(dataset)$ // $x, y = [S \times M, C]$
2: $datasetAE \leftarrow histCalc(datasetAE)$ // $x, y = [S \times M, H]$
3: $datasetAE \leftarrow norm(datasetAE)$
4: $train(autoEncoder, datasetAE, loss = \text{MSE})$
5: $lse \leftarrow merge(encoder, clf)$
6: $train(lse, dataset, loss = \text{crossEntropy})$

#### *1D Convolutional Neural Network Extractor (CNN1DE)*

The 1D Convolutional Neural Network Extractor (CNN1DE) branch consists of two, three, or four convolutional feature-extraction blocks, corresponding to small, medium, and large model variants, respectively. Each convolutional block comprises a 1D convolutional layer, a batch normalization layer, a LeakyReLU activation function, and a max-pooling layer that progressively reduces the temporal dimension.

Each CNN1DE variant additionally includes a classification block composed of a linear layer, a LeakyReLU activation, a second linear layer, and a sigmoid activation at the output. In the final convolutional block preceding the classifier, max pooling is omitted in order to preserve the resolution of the learned features.

The CNN1DE branch is trained independently on the same datasets as the LSE branch, using samples of shape $[num_cells, num_markers]$ and the cross-entropy loss for binary classification. The exact specification of the CNN1DE architecture, including the number of channels in each layer, kernel sizes, padding, and stride values, is provided in Appendix E.

**Algorithm 2** CNN1DE and CytoDNNEnsemble (CDE) training pipeline

**Require:** dataset // $x = [S, C, M]$, $y = [1]$
**Ensure:** trained CDE model // *Dimensions in comments: S – number of samples, C – number of cells, M – number of markers.*

1: $train(cnn1de, dataset, loss = \text{crossEntropy})$
2: $cde \leftarrow merge(lse, cnn1de, clf)$ // *pretrained LSE and CNN1DE*
3: $train(cde, dataset, loss = \text{crossEntropy})$

In summary, the proposed CytoDNNEnsemble (CDE) model operates on multi-cell flow cytometry samples represented as matrices of marker measurements and combines two complementary branches: CNN1DE for fine-grained, cell-level feature extraction and LSE for global, marker-level representation learning. Both branches are pretrained and then integrated into a parameter-efficient ensemble architecture, which is trained for sample-level binary classification using cross-entropy loss. The same data representations, preprocess-

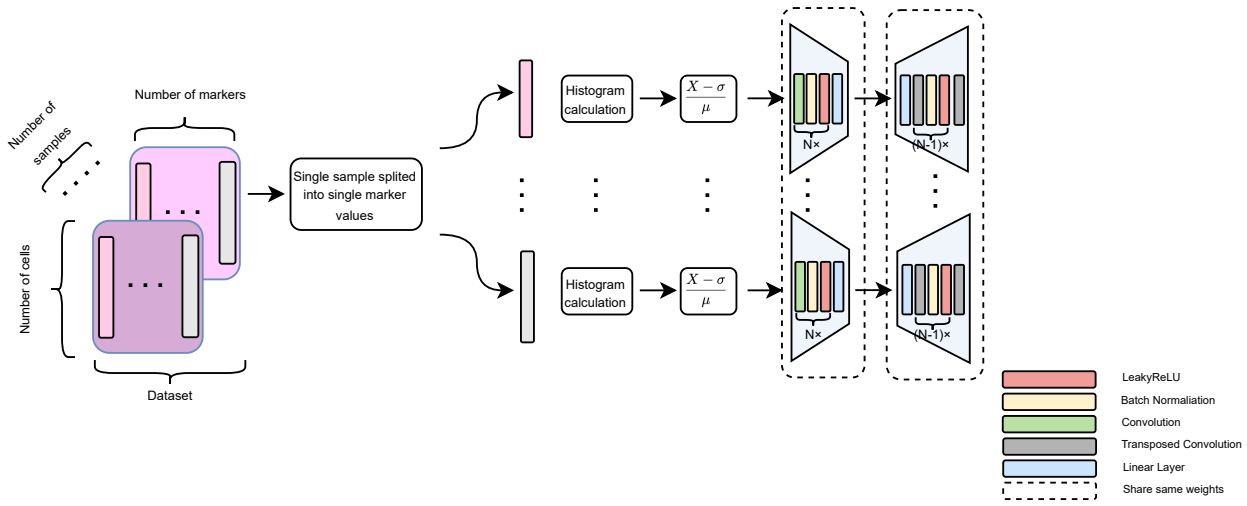



**Figure 2.** Autoencoder training scheme used in the Latent Space Extractor (LSE) branch. For simplicity, the scheme is shown for a single sample; in practice, all computations are performed in a batch-wise manner. During training, the values corresponding to successive markers are processed independently: their histograms are computed to reduce the dimensionality of the data and then normalized. A convolutional autoencoder is trained on the processed data to reconstruct the input using the mean squared error (MSE) loss. After training, the decoder is removed and the encoder is used as a feature extractor that maps normalized marker histograms into a low-dimensional latent space while preserving the most relevant information.

ing steps, and training/validation/test splits are used across all model variants to ensure fair and consistent evaluation.

## Experiments

As part of the experimental evaluation, the experiments described in[59] were reproduced using the CNN1DE, LSE, and CDE models and compared against baseline methods: CytoSet[59], CellCNN[60], and CytoDX[63]. The models were trained and evaluated on four flow and mass cytometry datasets: AML, HEUvsUE, HVTN, and NK-Cell, which are described above. The AML, HEUvsUE, and HVTN datasets were previously used in the FlowCAP-II benchmark challenge[15]. The NK-Cell dataset was used to evaluate the CellCNN model in[60].

Results are reported for two configurations of the CDE ensemble: CDE-medium and CDE-best. The CDE-medium configuration consists of the LSE-medium and CNN1DE-medium models, while the CDE-best configuration combines the best-performing LSE and CNN1DE models across all tested model sizes. As a result, the CDE-best configuration requires substantially more training time, as it involves training models of all sizes, whereas the CDE-medium configuration only requires training models of medium size. The choice between these configurations depends on task requirements: the CDE-medium model is faster to train and easier to deploy, while the CDE-best model demands more GPU resources but is expected to achieve superior performance.

### Evaluation metrics

We evaluate all models using accuracy and area under the ROC curve (AUC) on the AML, HEUvsUE, HVTN, and NK-Cell datasets. The resulting values are compared with those of existing methods from the literature, including CellCNN[60], CytoDX[63], and CytoSet[59]. In addition, precision, recall, and F1 scores are computed for all proposed models, with detailed results provided in Appendix C.

Accuracy is defined as

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN},$$

precision as

$$precision = \frac{TP}{TP+FP},$$

recall as

$$recall = \frac{TP}{TP+FN},$$

and the F1 score as

$$F1 = \frac{2TP}{2TP+FP+FN},$$

where

- $TP$ – true positives (positive samples predicted as positive),

- $TN$ – true negatives (negative samples predicted as negative),
- $FP$ – false positives (negative samples predicted as positive),
- $FN$ – false negatives (positive samples predicted as negative).

The AUC metric is defined as

$$AUC = \int_0^1 TPR(FPR)\, d(FPR),$$

where

- $TPR = \frac{TP}{TP+FN}$,
- $FPR = \frac{FP}{FP+TN}$.

To obtain reliable estimates, all models are evaluated using five-fold cross-validation[64] with fixed splits between training and evaluation samples and different random seeds. The evaluation protocol follows[59] to ensure a fair comparison between CytoDNNEnsemble and the baseline methods.

### Training configurations

We perform a grid search to identify suitable hyperparameters for the proposed method. The best-performing configuration is selected based on the highest validation performance under the five-fold evaluation described above. The hyperparameters considered include the learning rates for the LSE, CNN1DE, and CDE models; the learning rate of the autoencoder; and the latent dimensionality of the autoencoder, which forms part of the LSE branch. During training of the final classifier, the LSE and CNN1DE parameters can either be frozen (non-trainable) or unfrozen (trainable); this choice is also treated as a hyperparameter. In Figure 1, frozen connections are indicated by a snowflake symbol. Furthermore, for the final classifier, we search over the dimensionality of the hidden layer and the weight decay value to reduce the risk of overfitting.

The specific hyperparameter values used in the final configurations of CDE-medium and CDE-best are reported in Appendix G.

### Data preparation and subsampling

Classification of data originating from medical experiments is often challenging due to class imbalance: the majority of samples typically come from healthy individuals. In many medical applications, therefore, a weighted loss function is employed to prevent the model from always predicting the most frequent class and to increase the minority class.[65,66]. In our case, the AML, HEUvsUE, HVTN, and NK-Cell datasets also exhibit imbalance, with more samples from healthy individuals than from patients.

However, each sample contains a large number of cells (approximately 20,520 for AML, 440,384 for HEUvsUE, 199,285 for HVTN, and 13,080 for NK-Cell; see Table 1),

| Dataset | # class samples | | # samples | | # cells per sample (mean) | | # markers |
|---|---|---|---|---|---|---|---|
| | negative | positive | train | test | train | test | |
| AML | 316 | 43 | 286 | 73 | 20,468.7 | 20,719.2 | 40 |
| HEUvsUE | 140 | 168 | 246 | 62 | 445,024.7 | 421,969.6 | 11 |
| HVTN | 48 | 48 | 48 | 48 | 200,035.8 | 198,534.9 | 12 |
| NK-Cell | 11 | 9 | 14 | 6 | 12,311.0 | 14,873.2 | 43 |

**Table 1.** Characteristics of the datasets used in the experiments. The number of cells per sample is reported as the mean across all samples in each dataset.

which enables subsampling at the cell level. This strategy allows us to construct balanced training sets without modifying sample-level labels.

For a single sample of shape $[num_cells, num_markers]$, we can generate $\left\lfloor \frac{num_cells}{subsample_dim} \right\rfloor$ non-overlapping subsamples or a predefined number of overlapping subsamples. Using this procedure, we construct balanced datasets for all four benchmarks, each containing $2 \times 10^5$ subsamples per class (i.e., $2 \times 10^5$ positive and $2 \times 10^5$ negative subsamples). Subsamples are drawn as uniformly as possible across original samples, ensuring that each sample contributes a similar number of subsamples. As a result, the final training datasets for AML, HEUvsUE, HVTN, and NK-Cell have shape $[4 \times 10^5, n, num_markers]$, where $n$ denotes the number of cell vectors in a subsample and controls how many input cells are used by the model. To ensure a fair comparison, we follow Yi et al.[59] and use the same values of $n$ and the same subsampling procedure as in their work.

All raw marker values are rescaled using the inverse hyperbolic sine (arcsinh) transformation, parameterized by a scaling factor $c$:

$$\operatorname{arcsinh}\left(\frac{x}{c}\right) = \ln\left(\frac{x}{c} + \sqrt{\left(\frac{x}{c}\right)^2 + 1}\right),$$

with $c = 5$, following[59,67]. This transformation is standard in cytometry and reduces the impact of extreme values and skewed distributions.

### Optimizer and training schedule

We use the Adam optimizer[68] to train all proposed models, including CNN1DE, LSE, and CDE. The optimizer parameters $\beta_1$ and $\beta_2$ are set to 0.9 and 0.999, respectively, across all experiments. In addition, we employ the One-Cycle learning rate scheduler[69], which increases the learning rate from its initial value to a predefined maximum and then gradually decreases it to a minimum value substantially lower than the initial learning rate. Early stopping based on validation performance is applied during training to mitigate overfitting.

### Hardware and implementation details

All experiments were conducted on a workstation running Ubuntu 18.04.6 with 32 GB of RAM. Computations were performed on two GPUs: an NVIDIA GeForce GTX 1080 Ti with 11 GB of VRAM and an NVIDIA TITAN Xp with 12 GB

of VRAM. The models, training loops, and evaluation procedures were implemented in PyTorch[62], and data processing and transformations were performed using PyTorch and NumPy[70].

## Results

The experimental results for all datasets and sampling sizes are summarized in Tables 2–5. Overall, the proposed CytoDNNEnsemble (CDE) model achieves state-of-the-art performance on the two largest and most diverse datasets, AML and HEUvsUE, while remaining competitive on HVTN and NK-Cell. These findings support the hypothesis that combining local, cell-level and global, marker-level representations in a parameter-efficient ensemble improves robustness and diagnostic performance on multi-cell cytometry tasks.

### *AML dataset*

Table 2 reports the results for the AML dataset across different subsample sizes $n$. The CDE-best configuration achieves the highest accuracy for $n = 512$ and $n = 1024$, with accuracy values of 98.9% and AUC up to 99.9%, outperforming the strongest baseline (CellCNN or CytoSet, depending on the setting) by approximately 1–2.6 percentage points in accuracy and up to 1 percentage point in AUC. The CDE-medium configuration also consistently matches or exceeds the baselines, demonstrating that an ensemble composed of medium-size LSE and CNN1DE models is already sufficient to reach or surpass prior state-of-the-art performance.

Among the single models, LSE-large and CNN1DE-small provide strong improvements over previous methods, indicating that both the global (LSE) and local (CNN1DE) branches are effective on this dataset. The consistently high AUC values across all tested $n$ further confirm that the proposed architectures maintain excellent discriminative power as the number of sampled cells increases.

### *HEUvsUE dataset*

Table 3 presents the results for the HEUvsUE dataset. Here, the CDE-best ensemble provides the largest improvement over previous approaches. For $n = 1024, 2048$, and 4096, CDE-best improves accuracy by approximately $+25\%$, $+16\%$, and $+19\%$, respectively, compared with the best baseline (CellCNN or CytoSet), and increases AUC by approximately $+9\%$, $+16\%$, and $+12\%$ for the same values of $n$. These gains demonstrate the benefit of combining complementary local and global representations in a single ensemble.

Although individual LSE and CNN1DE models yield only moderate improvements over prior methods, their combination in the CDE-best consistently achieves the highest accuracy and AUC across all sampling sizes. This indicates that the ensemble can exploit the complementary strengths of both branches in this challenging data set.

### *HVTN and NK-Cell datasets*

Tables 4 and 5 show the results for the HVTN and NK-Cell datasets. On these smaller datasets, the proposed ensembles do not consistently outperform the best baselines. For HVTN, CytoSet remains the strongest method in terms of accuracy and AUC, although CDE-medium achieves competitive AUC values, especially for larger $n$. For NK-Cell, several baseline methods (CellCNN, CytoDX, CytoSet) achieve very high accuracy and AUC, leaving limited headroom for improvement.

Despite this, the proposed single models achieve robust performance: LSE-medium and LSE-large obtain accuracy and AUC values close to the best baselines on HVTN, and LSE-small performs best among the CDE branches on NK-Cell. These results suggest that, for datasets with a small number of patients (48 for HVTN and 14 for NK-Cell in the training sets), the capacity of the full ensemble may not be fully utilized, and a single branch (particularly LSE) can be a more appropriate choice.

### *Effect of subsampling and robustness across $n$*

An important observation from Tables 2 and 3, and Figures 10 and 11, is that the performance of previous models (CellCNN, CytoDX, CytoSet) often decreases as the number of sampled cells $n$ increases. In contrast, the proposed CDE model maintains a similar level of predictive performance across all values of $n$. This robustness is likely attributable to the LSE branch, which makes predictions based on normalized histograms of marker values and thus captures global, more stable features. As a result, CDE is less sensitive to the specific subsampling factor and yields more consistent performance across different configurations.

Figures 3 and 4 further summarize mean accuracy and AUC for all models across datasets as a function of model size. The proposed single models (CNN1DE and LSE) achieve robustness comparable to CytoSet in terms of accuracy, with minimum accuracy values around 65% and mean accuracy around 83%. For AUC, the minimum values are higher for CNN1DE and LSE (approximately 72% and 75%, respectively) than for CytoSet (around 66%), suggesting slightly improved robustness across sampling sizes.

### *Global comparison across models*

Figure 5 presents a global comparison of accuracy and AUC for all models, aggregated across all datasets and sampling values $n$. The proposed ensemble (CDE) achieves the highest minimum accuracy (approximately 77%) and AUC (approximately 85%), substantially exceeding the worst-case performance of baseline methods. In terms of average accuracy and AUC across all settings, CDE also achieves the best overall values, indicating that it is the most robust and generalizable model in this study.

### *Model size and parameter efficiency*

Tables 6 and 7 report the average number of parameters for medium and best model configurations, respectively. Because the number of parameters depends on the input dimensionality, values are reported separately for each dataset and averaged over all values of $n$. The proposed CDE method achieves comparable or superior performance to CytoSet while using significantly fewer parameters on three of the four bench-

| Category | Model | n = 256 | | n = 512 | | n = 1024 | |
|---|---|---|---|---|---|---|---|
| | | Accuracy | AUC | Accuracy | AUC | Accuracy | AUC |
| Other | CellCNN[60] | 98.1 | **99.8** | 96.7 | **99.9** | 95.6 | 98.5 |
| | CytoDX[63] | 94.0 | 98.3 | 94.3 | 98.4 | 94.8 | 98.6 |
| | CytoSet[59] | 97.0 | 99.1 | 95.9 | 99.3 | 95.4 | 98.8 |
| Single | LSE-small **(ours)** | 97.8 ± 0.67 | 99.7 ± 0.28 | 97.5 ± 1.03 | 99.6 ± 0.39 | 97.3 ± 0.87 | 99.7 ± 0.31 |
| | CNN1DE-small **(ours)** | 97.0 ± 1.60 | 96.3 ± 2.67 | 97.8 ± 0.67 | 97.7 ± 2.23 | 98.1 ± 1.10 | 99.7 ± 0.41 |
| | LSE-medium **(ours)** | **98.6 ± 0.87** | **99.8 ± 0.20** | 97.8 ± 1.10 | 99.8 ± 0.14 | 97.0 ± 1.03 | 99.8 ± 0.20 |
| | CNN1DE-medium **(ours)** | 95.6 ± 1.60 | 99.0 ± 1.02 | 95.9 ± 0.87 | 98.0 ± 2.42 | 96.4 ± 1.10 | 98.3 ± 2.17 |
| | LSE-large **(ours)** | 98.4 ± 1.34 | **99.8 ± 0.22** | 98.4 ± 0.55 | **99.9 ± 0.09** | 98.1 ± 1.40 | 99.8 ± 0.20 |
| | CNN1DE-large **(ours)** | 95.6 ± 2.01 | 97.1 ± 3.24 | 96.4 ± 1.40 | 95.7 ± 3.66 | 95.3 ± 1.64 | 95.8 ± 3.53 |
| Ensemble | CytoDNNEnsemble-medium **(ours)** | 97.5 ± 1.03 | 99.7 ± 0.19 | 98.1 ± 1.40 | 99.7 ± 0.39 | 97.5 ± 1.03 | 99.5 ± 0.20 |
| | CytoDNNEnsemble-best ❄**(ours)** | 98.4 ± 1.03 | 99.7 ± 0.48 | **98.9 ± 0.55** | 99.7 ± 0.26 | **98.9 ± 1.34** | **99.9 ± 0.14** |

**Table 2.** Classification performance on the AML dataset. Results are reported for baseline methods (CellCNN, CytoDX, CytoSet), single models (LSE and CNN1DE) and the proposed CytoDNNEnsemble (CDE) for different subsample sizes $n$. Underlined values indicate the best result within each model category (Other, Single, Ensemble), while bold values indicate the best result across all models. Single models used in the CDE-best ensemble are marked with a wavy underline. CDE-medium combines LSE and CNN1DE in medium size; the snowflake symbol (❄) indicates that extractor weights are frozen during training of the final classifier.

| Category | Model | n = 1024 | | n = 2048 | | n = 4096 | |
|---|---|---|---|---|---|---|---|
| | | Accuracy | AUC | Accuracy | AUC | Accuracy | AUC |
| Other | CellCNN[60] | 56.5 | 79.0 | 56.5 | 75.4 | 63.8 | 75.4 |
| | CytoDX[63] | 43.2 | 43.8 | 43.2 | 43.5 | 42.9 | 44.1 |
| | CytoSet[59] | 64.4 | 65.5 | 68.0 | 71.8 | 65.7 | 73.8 |
| Single | LSE-small **(ours)** | 66.1 ± 4.08 | 74.5 ± 4.36 | 67.4 ± 2.58 | 77.8 ± 2.37 | 71.6 ± 4.52 | 81.5 ± 3.28 |
| | CNN1DE-small **(ours)** | 56.5 ± 5.95 | 56.5 ± 6.20 | 53.6 ± 2.78 | 50.4 ± 2.98 | 52.3 ± 4.85 | 55.4 ± 6.64 |
| | LSE-medium **(ours)** | 68.1 ± 4.72 | 75.2 ± 4.37 | 67.1 ± 3.47 | 76.2 ± 2.85 | 69.7 ± 4.61 | 79.6 ± 3.40 |
| | CNN1DE-medium **(ours)** | 63.9 ± 2.41 | 70.3 ± 5.00 | 62.3 ± 3.32 | 70.2 ± 3.20 | 60.0 ± 3.44 | 66.0 ± 4.29 |
| | LSE-large **(ours)** | 67.1 ± 3.76 | 74.6 ± 4.32 | 67.4 ± 3.13 | 78.8 ± 2.94 | 70.3 ± 2.19 | 80.9 ± 3.68 |
| | CNN1DE-large **(ours)** | 70.3 ± 4.74 | 78.7 ± 6.32 | 69.7 ± 7.59 | 78.2 ± 6.86 | 65.5 ± 8.63 | 71.7 ± 6.11 |
| Ensemble | CytoDNNEnsemble-medium **(ours)** | 71.6 ± 5.46 | 80.8 ± 4.23 | 69.7 ± 5.53 | 76.0 ± 4.87 | 66.5 ± 4.83 | 74.5 ± 6.53 |
| | CytoDNNEnsemble-best **(ours)** | **80.7 ± 4.21** | **86.1 ± 3.94** | **79.0 ± 3.95** | **87.7 ± 2.47** | **78.4 ± 3.90** | **84.6 ± 4.33** |

**Table 3.** Classification performance on the HEUvsUE dataset. Results are reported for baseline methods, single models, and the proposed ensembles for different subsample sizes $n$. Underlined values indicate the best result within each model category, while bold values indicate the best result overall. Single models used in CDE-best are marked with a wavy underline.

| Category | Model | n = 256 | | n = 512 | | n = 1024 | |
|---|---|---|---|---|---|---|---|
| | | Accuracy | AUC | Accuracy | AUC | Accuracy | AUC |
| Other | CellCNN[60] | 82.9 | 93.4 | 79.6 | 92.7 | 83.7 | 93.3 |
| | CytoDX[63] | 65.8 | 74.6 | 62.9 | 73.8 | 65.4 | 74.1 |
| | CytoSet[59] | **91.3** | **96.3** | **89.2** | **96.4** | **88.3** | **95.7** |
| Single | LSE-small **(ours)** | 81.3 ± 1.86 | 87.0 ± 0.48 | 80.4 ± 2.13 | 89.1 ± 1.29 | 85.4 ± 2.64 | 90.8 ± 2.78 |
| | CNN1DE-small **(ours)** | 74.2 ± 3.12 | 82.9 ± 2.61 | 73.3 ± 3.58 | 80.6 ± 4.11 | 62.9 ± 6.64 | 79.7 ± 2.76 |
| | LSE-medium **(ours)** | 81.7 ± 1.56 | 87.2 ± 0.65 | 80.8 ± 0.83 | 89.6 ± 1.11 | 86.3 ± 2.13 | 91.1 ± 2.36 |
| | CNN1DE-medium **(ours)** | 71.3 ± 4.25 | 81.5 ± 3.03 | 73.8 ± 4.86 | 86.7 ± 3.09 | 71.7 ± 4.86 | 85.7 ± 1.72 |
| | LSE-large **(ours)** | 81.3 ± 1.86 | 86.8 ± 0.63 | 82.5 ± 2.50 | 90.0 ± 1.23 | 85.8 ± 2.04 | 90.6 ± 2.27 |
| | CNN1DE-large **(ours)** | 75.0 ± 2.95 | 85.1 ± 1.66 | 77.1 ± 4.75 | 87.5 ± 3.64 | 73.3 ± 4.82 | 83.8 ± 2.77 |
| Ensemble | CytoDNNEnsemble-medium ❄**(ours)** | 81.7 ± 3.33 | 88.9 ± 2.35 | 80.4 ± 2.83 | 92.3 ± 1.63 | 85.8 ± 5.65 | 94.2 ± 2.23 |
| | CytoDNNEnsemble-best ❄**(ours)** | 81.7 ± 2.43 | 88.8 ± 3.26 | 82.5 ± 2.13 | 92.3 ± 1.74 | 84.2 ± 5.83 | 92.2 ± 2.86 |

**Table 4.** Classification performance on the HVTN dataset. Underlined values indicate the best result within each model category; bold values indicate the best result among all models. Single models that are used in the final CDE-best model are marked in wavy underline. CDE-medium model consists of the LSE and CNN1DE models in medium sizes. The snowflake symbol (❄) indicates that the CDE model has frozen single extractor weights during training.

mark datasets, particularly for HEUvsUE and HVTN, where parameter reductions exceed 60%.

Notably, in several cases the LSE-best single model alone achieves performance comparable to or better than CytoSet

| Category | Model | n = 256 | | n = 512 | | n = 1024 | |
|---|---|---|---|---|---|---|---|
| | | Accuracy | AUC | Accuracy | AUC | Accuracy | AUC |
| Other | CellCNN[60] | **83.3** | 95.0 | 76.7 | 97.5 | 66.7 | 82.5 |
| | CytoDX[63] | **83.3** | 92.5 | **83.3** | **100.0** | 83.3 | 85.0 |
| | CytoSet[59] | 80.0 | **97.5** | **83.3** | 97.5 | **86.7** | 95.0 |
| Single | LSE-small **(ours)** | 73.3 ± 8.17 | 90.0 ± 9.35 | **83.3 ± 0.00** | 97.5 ± 5.00 | 83.3 ± 0.00 | 97.5 ± 5.00 |
| | CNN1DE-small **(ours)** | 73.3 ± 13.3 | 92.5 ± 10.0 | 80.0 ± 6.67 | 92.5 ± 6.12 | 83.3 ± 0.00 | 85.0 ± 5.00 |
| | LSE-medium **(ours)** | 70.0 ± 12.5 | 92.5 ± 10.0 | **83.3 ± 0.00** | 92.5 ± 6.12 | 83.3 ± 0.00 | 97.5 ± 5.00 |
| | CNN1DE-medium **(ours)** | 73.3 ± 8.17 | 90.0 ± 5.00 | 80.0 ± 6.67 | 90.0 ± 5.00 | 80.0 ± 6.67 | 87.5 ± 7.91 |
| | LSE-large **(ours)** | 73.3 ± 13.3 | 87.5 ± 11.2 | **83.3 ± 0.00** | 90.0 ± 5.00 | **86.7 ± 6.67** | **100 ± 0.00** |
| | CNN1DE-large **(ours)** | 63.3 ± 6.67 | 90.0 ± 12.3 | 70.0 ± 12.5 | 92.5 ± 10.0 | 73.3 ± 8.17 | 87.5 ± 0.00 |
| Ensemble | CytoDNNEnsemble-medium ❄**(ours)** | 76.7 ± 8.17 | 95.0 ± 10.0 | 80.0 ± 6.67 | **100 ± 0.00** | 80.0 ± 6.67 | 95.0 ± 6.12 |
| | CytoDNNEnsemble-best **(ours)** | 76.7 ± 8.17 | **97.5 ± 5.00** | 80.0 ± 6.67 | 90.0 ± 9.35 | 83.3 ± 0.00 | 92.5 ± 6.12 |

**Table 5.** Classification performance on the NK-Cell dataset. Underlined values indicate the best result within each model category; bold values indicate the best result among all models. Single models that are used in the final CDE-best model are marked in wavy underline. CDE-medium model consists of the LSE and CNN1DE models in medium sizes. The snowflake symbol (❄) indicates that the CDE model has frozen single extractor weights during training.

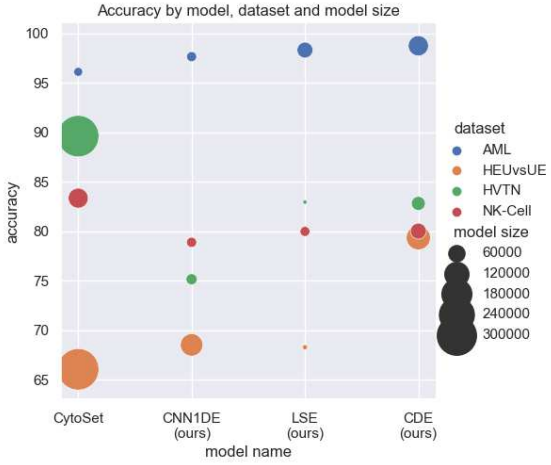



**Figure 3.** Mean accuracy for each model across all subsample sizes $n$ and datasets. The diameter of each circle represents the average model size (number of parameters) for a given model and dataset.

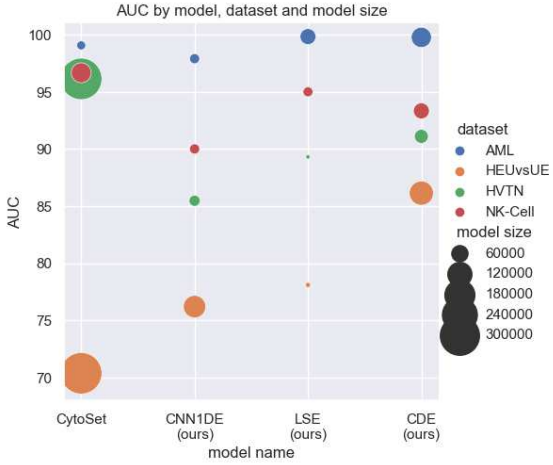



**Figure 4.** Mean AUC for each model across all subsample sizes $n$ and datasets. The diameter of each circle represents the average model size (number of parameters) for a given model and dataset.

| Dataset name | CytoSet model size | LSE model size-medium (ours) | CNN1DE model size-medium (ours) | CDE model size-medium (ours) |
|---|---|---|---|---|
| AML | 25,217 | 27,785 | 30,622.3 | 58,416.3 (+131.65%) |
| HEUvsUE | 334,081 | 25,257 | 98,718.3 | 124,008.3 (-62.88%) |
| HVTN | 334,593 | 13,449 | 27,038.3 | 40,496.3 (-87.90%) |
| NK-Cell | 85,249 | 29,321 | 31,134.3 | 60,464.3 (-29.07%) |

**Table 6.** Average number of parameters for medium-size models. The percentage change in parameters for the proposed CDE-medium model relative to CytoSet is shown in parentheses (red for an increase, green for a reduction).

| Dataset name | CytoSet model size | LSE model size-best (ours) | CNN1DE model size-best (ours) | CDE model size-best (ours) |
|---|---|---|---|---|
| AML | 25,217 | 57,609 | 29,246.3 | 86,864.3 (+236.54%) |
| HEUvsUE | 334,081 | 15,081 | 105,054.3 | 120,152.3 (-64.04%) |
| HVTN | 334,593 | 13,449 | 33,374.3 | 46,832.3 (-90.03%) |
| NK-Cell | 85,249 | 28,633 | 29,502.3 | 58,144.3 (-31.79%) |

**Table 7.** Average number of parameters for best-performing models. The percentage change in parameters for CDE-best relative to CytoSet is shown in parentheses (red for an increase, green for a reduction).

while using substantially fewer parameters: approximately 22 times fewer parameters on HEUvsUE, 25 times fewer on HVTN, and 3 times fewer on NK-Cell. For AML, LSE uses roughly twice as many parameters as CytoSet but still achieves competitive or superior accuracy and AUC (see Table 7). Overall, these results indicate that the proposed architectures, and in particular the CDE ensemble, offer a favorable trade-off between predictive performance and model size, making them suitable for deployment on hardware with limited VRAM or in resource-constrained settings.

***Summary of results and practical implications***

Across all datasets and sampling sizes, no single baseline method (CellCNN, CytoDX, CytoSet) dominates in every

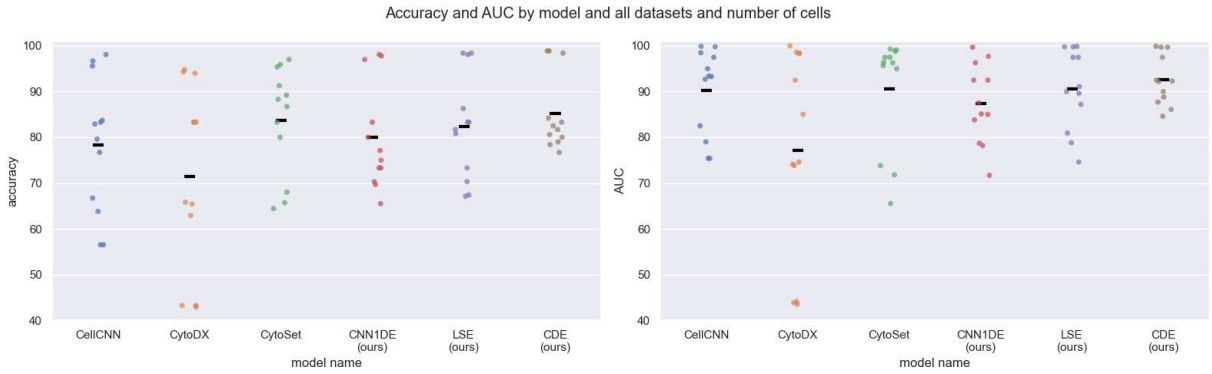



**Figure 5.** Accuracy and AUC values for baseline models (CellCNN, CytoDX, CytoSet), single models (CNN1DE, LSE), and the proposed CytoDNNEnsemble (CDE) across all datasets and subsample sizes $n$. The black line indicates the average metric value for each model.

scenario, highlighting the inherent difficulty of multi-cell cytometry classification. A similar pattern is observed among the individual models within the CDE framework: depending on the dataset, different model sizes and branches (LSE or CNN1DE) emerge as the best-performing single model. This variability supports the use of ensembles that can combine multiple specialized components.

The proposed CytoDNNEnsemble leverages this diversity and consistently provides the best overall robustness and average performance across datasets, while remaining parameter-efficient on most benchmarks. In practice, the modular design of the framework allows flexible deployment: LSE or CNN1DE can be used independently when computational resources are limited, and the full CDE ensemble can be activated when maximum predictive performance is required. A reference implementation, including training and evaluation scripts and hyperparameter search utilities, is publicly available in our GitHub repository[71].

## Conclusion

In this work, we introduced CytoDNNEnsemble, a parameter-efficient deep ensemble architecture for flow and mass cytometry-based diagnosis. The model combines two complementary branches: CNN1DE, which focuses on fine-grained, cell-level patterns, and LSE, which captures global, marker-level trends from normalized histograms. By integrating these local and global representations, CytoDNNEnsemble is able to learn rich sample-level features from high-dimensional cytometry data.

The proposed ensemble achieves state-of-the-art performance on the two largest and most diverse benchmark datasets, AML and HEUvsUE, while remaining competitive on the smaller HVTN and NK-Cell datasets. These results suggest that CytoDNNEnsemble is particularly well suited to settings where data are available from a larger number of individuals, allowing the model to fully exploit its dual-branch design. On smaller datasets, single-branch models—especially the LSE variant—already provide strong performance with fewer parameters, indicating that model capacity should be matched to the amount of available data.

From a practical perspective, CytoDNNEnsemble is designed to be computationally efficient and easy to deploy. All experiments were conducted on standard consumer GPUs with less than 12 GB of VRAM, demonstrating that the proposed models do not require a specialized high-performance computing infrastructure. The modular structure of the framework allows LSE and CNN1DE to be used independently or combined in the full ensemble, enabling flexible deployment strategies depending on available computational resources and desired accuracy.

In general, the combination of robust predictive performance in large-scale datasets, favorable parameter efficiency, and modular design positions CytoDNNEnsemble as a versatile tool for cytometry-based analysis and diagnostic decision support. Future work will focus on extending the framework to multi-class problems, integrating additional sources of clinical information, and investigating strategies to improve performance and calibration on small-cohort datasets.

## Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

## Author contributions statement

**T.G.**: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft; **U.M.-K.**: Conceptualization, Formal analysis, Methodology, Project administration, Resources, Supervision, Writing – review and editing.

## Data availability

All datasets used to perform the described experiments are publicly available[72].

## Additional information

## References


1. Kim, M. et al. Biomed. Eng., Biomed. Inform. Tech., chap. Deep learning in biomedical image analysis, 239–263 (Academic Press, https://doi.org/10.1016/B978-0-12-816034-3.00008-0., 2020).

2. Baldi, P. Deep learning in biomedical data science. Annu. Rev. Biomed. Data Sci. **1**, 181–205, DOI: https://doi.org/10.1146/annurev-biodatasci-080917-013343 (2018).

3. Zhang, Y., Dong, Z., Li, S. & Cattani, C. Deep learning methods for biomedical information analysis. J. Ambient Intell. Humaniz. Comput. **14**, 5293—5296 (2023).

4. Zemouri, R., Zerhouni, N. & Racoceanu, D. Deep learning in the biomedical applications: Recent and future status. Appl. Sci. **9**, 1526, DOI: 10.3390/app9081526 (2019).

5. Yue, A. et al. Ai in flow cytometry: Current applications and future directions. Cytom. Part B: Clin. Cytom. **108**, 404–420, DOI: 10.1002/cyto.b.22255 (2025).

6. Li, Y. et al. Deep cytometry: Deep learning with real-time inference in cell sorting and flow cytometry. Sci. Reports **9**, 11088, DOI: 10.1038/s41598-019-47193-6 (2019).

7. Zhou, J. et al. Multi-model deep learning on photoacoustic flow cytometry signals for real-time melanoma circulating tumor cells detection and biological characterization. Expert. Syst. with Appl. **308**, 131123, DOI: https://doi.org/10.1016/j.eswa.2026.131123 (2026).

8. Gil, F., Osowski, S., Swiderski, B. & Słowińska, M. Ensemble of classifiers based on deep learning for medical image recognition. Metrol. Meas. Syst. **30**, DOI: 10.24425/mms.2023.144400 (2023).

9. Deshmukh, P. K. Improving medical image classification using ensemble learning and deep convolutional neural networks. Int. J. Intell. Syst. Appl. Eng. **12**, 106–121 (2023).

10. Dang, T., Nguyen, T. T., McCall, J. A. W., Elyan, E. & Moreno-García, C. F. Two-layer ensemble of deep learning models for medical image segmentation. Cogn. Comput. **16**, 1141–1160 (2024).

11. Pi, S.-W., Lee, B.-D., Lee, M. S. & Lee, H. J. Ensemble deep-learning networks for automated osteoarthritis grading in knee x-ray images. Sci. Reports **13**, 22887 (2023).

12. Aniol, A., Pietron, M. & Duda, J. Ensemble approach for natural language question answering problem. In CANDARW 2019, 180–183, DOI: 10.1109/CANDARW.2019.00039 (2019).

13. Rane, N., Choudhary, S. P. & Rane, J. Ensemble deep learning and machine learning: applications, opportunities, challenges, and future directions. Stud. Med. Heal. Sci. **1**, 18–41, DOI: 10.48185/smhs.v1i2.1225 (2024).

14. Mishra, A., Bisoy, S. K., Kollie Jr, D. S., Priyadarshi, R. & Ranjan, R. An ensemble-based sentiment analysis approach for precision medicine recommendation. Sci. Reports (2026).

15. Aghaeepour, N. et al. Critical assessment of automated flow cytometry data analysis techniques. Nat. methods **10**, 228–238 (2013).

16. Commenges, D., Alkhassim, C., Gottardo, R., Hejblum, B. & Thiébaut, R. cytometree: A binary tree algorithm for automatic gating in cytometry analysis. Cytom. Part A **93**, 1132–1140, DOI: 10.1002/cyto.a.23601 (2018).

17. Gelbard, R. B. et al. A random forest model using flow cytometry data identifies pulmonary infection after thoracic injury. The J. Trauma Acute Care Surg. **95**, 39–46, DOI: 10.1097/TA.0000000000003937 (2023).

18. Cordeiro, E. M., Wanderley, B. M. S., Araújo, D. S. A. d. & Neto, A. D. D. Autogating in flow cytometry data using svm classifiers for bacterioplankton identification. In CSCI 2017, 1265–1269, DOI: 10.1109/CSCI.2017.222 (2017).

19. Wong, N., Kim, D., Robinson, Z., Huang, C. & Conboy, I. M. K-means quantization for a web-based open-source flow cytometry analysis platform. Sci. Reports **11**, 6735, DOI: 10.1038/s41598-021-85964-2 (2021).

20. Ye, X. & Ho, J. W. K. Ultrafast clustering of single-cell flow cytometry data using flowgrid. BMC Syst. Biol. **13**, 35, DOI: 10.1186/s12918-019-0726-3 (2019).

21. Rubbens, P., Props, R., Kerckhof, F.-M., Boon, N. & Waegeman, W. Phenogmm: Gaussian mixture modeling of cytometry data quantifies changes in microbial community structure. mSphere **6**, e00530–20, DOI: 10.1128/mSphere.00530-20 (2021).

22. Van Gassen, S. et al. Flowsom: Using self-organizing maps for visualization and interpretation of cytometry data. Cytom. Part A **87**, 636–645, DOI: 10.1002/cyto.a.22625 (2015).

23. Ujas, T. A., Mehta, M., Comi, T. J., Wilkerson, E. M. & Ko, D. C. A guide on analyzing flow cytometry data using clustering methods and nonlinear dimensionality reduction (tsne or umap). Methods Mol. Biol. **2616**, 231–249, DOI: 10.1007/978-1-0716-2926-0_18 (2023).

24. Spies, N. C. et al. Machine learning methods in clinical flow cytometry. Cancers **17**, 483, DOI: 10.3390/cancers17030483 (2025).

25. He, K. et al. Masked autoencoders are scalable vision learners. In CVPR 2022, 16000–16009, DOI: 10.1109/CVPR52688.2022.01552 (2022).

**26.** Saeys, Y., Van Gassen, S. & Lambrecht, B. N. Computational flow cytometry: helping to make sense of high-dimensional immunology data. Nat. Rev. Immunol. **16**, 449–462, DOI: 10.1038/nri.2016.56 (2016).

**27.** Mashford, B. S., Hewitt, T., May, M., Chuah, A. & Andrews, D. Comparison of deep-learning models for classification of cellular phenotype from flow cytometry data. IEEE Transactions on Comput. Biol. Bioinforma. 1–7, DOI: 10.1109/TCBBIO.2025.3562597 (2025).

**28.** Bank, D., Koenigstein, N. & Giryes, R. Autoencoders. In Rokach, L., Maimon, O. & Shmueli, E. (eds.) Machine Learning for Data Science Handbook, 353–374, DOI: 10.1007/978-3-031-24628-9_16 (Springer, 2023).

**29.** Zhang, Z., Chen, X., Tang, R. et al. Interpretable unsupervised learning enables accurate clustering with high-throughput imaging flow cytometry. Sci. Reports **13**, 20533, DOI: 10.1038/s41598-023-46782-w (2023).

**30.** Kim, J. et al. Cytometry masked autoencoder: An accurate and interpretable automated immunophenotyper. Cell Reports Medicine **5**, 101808, DOI: 10.1016/j.xcrm.2024.101808 (2024).

**31.** Phuycharoen, M., Kaestele, V., Williams, T. & Rattray, M. Uvae: Integration of unpaired and heterogeneous clinical flow cytometry data using an unbiasing variational autoencoder. bioRxiv (preprint) (2023). Preprint. Full text available at ResearchGate.

**32.** Kingma, D. P. & Welling, M. Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2022). Version 7, updated December 10, 2022.

**33.** Inecik, K., Meric, A., König, L. & Theis, F. J. flowvi: Flow cytometry variational inference. bioRxiv DOI: 10.1101/2023.11.10.566661 (2023). Preprint.

**34.** Grill, J.-B. et al. Bootstrap your own latent: A new approach to self-supervised learning. In Advances in Neural Information Processing Systems, vol. 33, 21271–21284 (2020).

**35.** Chen, T., Kornblith, S., Norouzi, M. & Hinton, G. A simple framework for contrastive learning of visual representations. In ICML 2020, vol. 119, 1597–1607 (PMLR, 2020).

**36.** Oquab, M. et al. DINOv2: Learning robust visual features without supervision. Transactions on Mach. Learn. Res. (2024). Featured Certification.

**37.** Hong, T., Peng, M., Kim, Y., Schellhorn, H. E. & Fang, Q. Automated cell profiling in imaging flow cytometry with annotation-efficient learning. Opt. & Laser Technol. **181**, 111992, DOI: 10.1016/j.optlastec.2024.111992 (2025).

**38.** Wu, Z. et al. Dynamorph: self-supervised learning of morphodynamic states of live cells. Mol. Biol. Cell **33**, DOI: 10.1091/mbc.E21-10-0511 (2022).

**39.** Huang, Q. et al. Imaging flow cytometry: from high-resolution morphological imaging to innovation in high-throughput multidimensional biomedical analysis. Front. Bioeng. Biotechnol. **13**, DOI: 10.3389/fbioe.2025.1580749 (2025).

**40.** Zhang, Z. et al. Predicting cell properties with ai from 3d imaging flow cytometer data. Sci. Reports **15**, 5715, DOI: 10.1038/s41598-024-80722-6 (2025).

**41.** He, K., Zhang, X., Ren, S. & Sun, J. Deep residual learning for image recognition. In CVPR 2016, 770–778, DOI: 10.1109/CVPR.2016.90 (IEEE, 2016).

**42.** Tan, M. & Le, Q. V. Efficientnet: Rethinking model scaling for convolutional neural networks. In ICML 2019, 6105–6114 (PMLR, 2019).

**43.** Dosovitskiy, A. et al. An image is worth 16x16 words: Transformers for image recognition at scale. Int. Conf. on Learn. Represent. (ICLR) (2021).

**44.** Eulenberg, P. et al. Reconstructing cell cycle and disease progression using deep learning. Nat. Commun. **8**, 463, DOI: 10.1038/s41467-017-00439-5 (2017).

**45.** Suffian, M. et al. Machine learning for automated gating of flow cytometry data. In CEUR Workshop Proceedings, vol. 3307, 47–56 (Sun SITE Central Europe, RWTH Aachen University, 2022).

**46.** Arik, S. Ö. & Pfister, T. Tabnet: Attentive interpretable tabular learning. In AAAI 2021, vol. 35, 6679–6687 (2021).

**47.** Wödlinger, M. et al. Automated identification of cell populations in flow cytometry data with transformers. Comput. Biol. Medicine **144**, 105314, DOI: 10.1016/j.compbiomed.2022.105314 (2022).

**48.** Spies, N. C. et al. Machine learning methods in clinical flow cytometry. Cancers (Basel) **17**, 483, DOI: 10.3390/cancers17030483 (2025).

**49.** Bierzanowski, S. & Pietruczuk, K. Revolution in flow cytometry: using artificial intelligence for data processing and interpretation. Eur. J. Transl. Clin. Medicine **8**, 83–96, DOI: 10.31373/ejtcm/199793 (2025).

**50.** Boushehri, S. S. et al. Explainable machine learning for profiling the immunological synapse and functional characterization of therapeutic antibodies. Nat. Commun. **14**, 7888, DOI: 10.1038/s41467-023-43429-2 (2023).

**51.** Ultsch, A. et al. An explainable ai system for the diagnosis of high-dimensional biomedical data. BioMedInformatics **4**, 13, DOI: 10.3390/biomedinformatics4010013 (2021).

**52.** Zuromski, L. M. et al. Clinical validation of a real-time machine learning-based system for the detection of acute myeloid leukemia by flow cytometry. Cytom. Part B: Clin. Cytom. DOI: 10.1002/cyto.b.22229 (2024). Epub ahead of print.

**53.** Alotaibi, A. Ensemble deep learning approaches in health care: A review. Comput. Mater. Continua **82**, 3741–3771, DOI: https://doi.org/10.32604/cmc.2025.061998 (2025).

**54.** Sorrentino, S. et al. Deep ensemble learning and transfer learning methods for classification of senescent cells from nonlinear optical microscopy images. Front. Chem. **Volume 11 - 2023**, DOI: 10.3389/fchem.2023.1213981 (2023).

**55.** Gupta, A. et al. Deep learning in image cytometry: A review. Cytom. Part A **95**, 366–380, DOI: 10.1002/cyto.a.23701 (2019).

**56.** Seoni, S. et al. Application of uncertainty quantification to artificial intelligence in healthcare: A review of last decade (2013–2023). Comput. Biol. Medicine **165**, 107441, DOI: https://doi.org/10.1016/j.compbiomed.2023.107441 (2023).

**57.** Balendran, A. et al. A scoping review of robustness concepts for machine learning in healthcare. NPJ Digit. Medicine **8**, 38, DOI: 10.1038/s41746-024-01420-1 (2025).

**58.** Spearman, P. et al. A trimeric, v2-deleted hiv-1 envelope glycoprotein vaccine elicits potent neutralizing antibodies but limited breadth of neutralization in human volunteers. J. Infect. Dis. **203**, 1165–1173 (2011).

**59.** Yi, H. & Stanley, N. Cytoset: Predicting clinical outcomes via set-modeling of cytometry data. In BCB 2021, 1–8 (2021).

**60.** Arvaniti, E. & Claassen, M. Sensitive detection of rare disease-associated cell subsets via representation learning. Nat. communications **8**, 14825 (2017).

**61.** Xu, B., Wang, N., Chen, T. & Li, M. Empirical evaluation of rectified activations in convolutional network. arXiv preprint arXiv:1505.00853 (2015).

**62.** Paszke, A. et al. Pytorch: an imperative style, high-performance deep learning library. In NIPS 2019 (Curran Associates Inc., Red Hook, NY, USA, 2019).

**63.** Hu, Z., Glicksberg, B. S. & Butte, A. J. Robust prediction of clinical outcomes using cytometry data. Bioinformatics **35**, 1197–1203, DOI: 10.1093/bioinformatics/bty768 (2019).

**64.** Bradshaw, T. J., Huemann, Z., Hu, J. & Rahmim, A. A guide to cross-validation for artificial intelligence in medical imaging. Radiol. Artif. Intell. **5**, e220232, DOI: 10.1148/ryai.220232 (2023).

**65.** Scholz, D. et al. Imbalance-aware loss functions improve medical image classification. In Burgos, N. et al. (eds.) MIDL 2024, vol. 250 of Proceedings of Machine Learning Research, 1341–1356 (PMLR, 2024).

**66.** Nugroho, B. A. An aggregate method for thorax diseases classification. Sci. Reports **11**, 3242 (2021).

**67.** Azad, A., Rajwa, B. & Pothen, A. flowvs: channel-specific variance stabilization in flow cytometry. BMC bioinformatics **17**, 1–14 (2016).

**68.** Kingma, D. P. & Ba, J. Adam: A method for stochastic optimization. In ICLR 2015 (2015).

**69.** Smith, L. N. & Topin, N. Super-convergence: Very fast training of neural networks using large learning rates. In Art. intell. and mach. learn. for multi-domain oper. applic., vol. 11006, 369–386 (SPIE, 2019).

**70.** Harris, C. R. et al. Array programming with NumPy. Nature **585**, 357–362, DOI: 10.1038/s41586-020-2649-2 (2020).

**71.** Gniazdowski, T. Cytodnnensemble–code (2026). Code implementation accompanying related cytometry classification research https://github.com/TomekGniazdowski/CytoDNNEnsemble-code.

**72.** Yi, H. & Stanley, N. Dataset–cytoset: Predicting clinical outcomes via set-modeling of cytometry data (2021). Dataset associated with Yi and Stanley, BCB 2021 paper "CytoSet: Predicting clinical outcomes via set-modeling of cytometry data" https://drive.google.com/drive/folders/1chfsOYSCsRg7kcydqyrze6B1Sg9-iInL?usp=share_link.


## A All model sizes

Table 8 reports the average number of parameters for all model size configurations. The latent dimensionality of the LSE model is fixed at 64 in all settings.

| Size | Model | AML | HEUvsUE | HVTN | NK-Cell |
|---|---|---|---|---|---|
| - | CytoSet | 25,217 | 334,081 | 334,593 | 85,249 |
| Small | LSE **(ours)** | 54,009 | 24,313 | 25,337 | 57,081 |
| | CNN1DE **(ours)** | 29,246.3 | 97,086.3 | 25,662.3 | 29,502.3 |
| Medium | LSE **(ours)** | 54,953 | 25,257 | 26,281 | 58,025 |
| | CNN1DE | 30,622.3 | 98,718.3 | 27,038.3 | 31,134.3 |
| | CDE **(ours)** | 85,584.3 | 124,008.3 | 53,328.3 | 89,168.3 |
| Large | LSE **(ours)** | 57,609 | 27,913 | 28,937 | 60,681 |
| | CNN1DE **(ours)** | 36,446.3 | 105,054.3 | 33,374.3 | 37,470.3 |

**Table 8.** Average number of model parameters (for all size configurations, with the LSE model latent dimensionality fixed at 64). Since the number of parameters depends on the input dimensionality, values are reported per dataset and averaged across all sampled cell counts ($n$).

## B Single model predictions

Figures 6, 7, 8, and 9 show the predictions of the single models (LSE and CNN1DE) alongside the final CDE ensemble model for different values of the sampling parameter $n$. In most cases, the ensemble (CDE), which combines the outputs of LSE and CNN1DE, is able to correct errors when one branch predicts correctly and the other does not, illustrating the complementary behavior of the two single models.

Predictions on the HEUvsUE dataset (Figure 7) further illustrate that the single models frequently produce different outputs. HEUvsUE is considered the most challenging dataset among the analyzed benchmarks and was also identified as the most difficult dataset in the FlowCAP-II challenge. The discrepancies between the single-model predictions support this assessment and highlight the value of analyzing samples from complementary perspectives—global with LSE and local with CNN1DE. In multiple cases, the ensemble successfully corrects errors made by individual models, confirming that this dual-perspective design is effective for improving prediction accuracy on complex cytometry datasets.

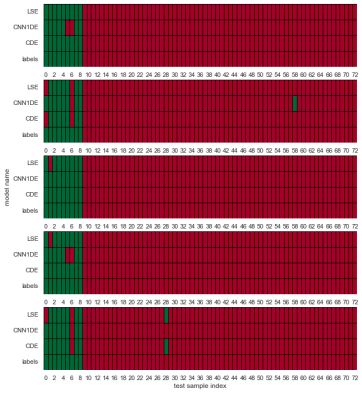



**Figure 6.** Predictions of proposed models: LSE, CNN1DE and the ensemble model - CDE for the AML dataset along with their corresponding labels. A red cell indicates a negative class, a green cell indicates a positive class.

## C Precision, recall, F1 metric values

Tables 9, 10, 11, and 12 report the precision, recall, and F1 scores for both the individual models and the ensemble. In most cases, using the ensemble either improves these metrics or leaves them unchanged compared to the corresponding single models.

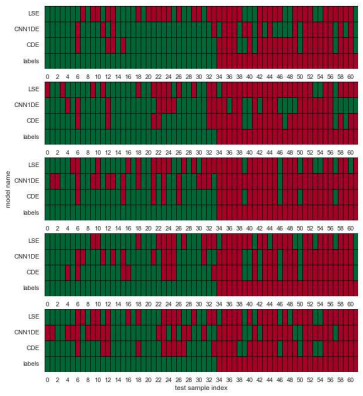



**Figure 7.** Predictions of proposed models: LSE, CNN1DE and the ensemble model - CDE for the HEUvsUE dataset along with their corresponding labels. A red cell indicates a negative class, a green cell indicates a positive class.

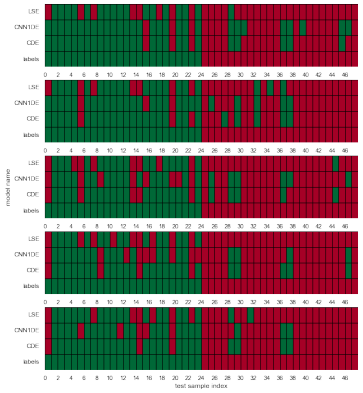



**Figure 8.** Predictions of proposed models: LSE, CNN1DE and the ensemble model - CDE for the HVTN dataset along with their corresponding labels. A red cell indicates a negative class, a green cell indicates a positive class.

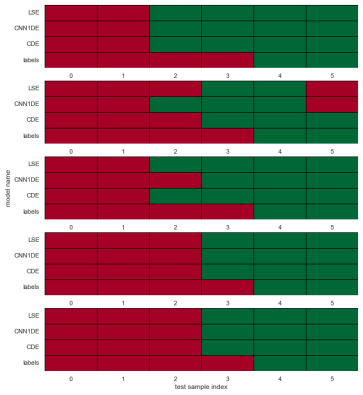



**Figure 9.** Predictions of proposed models: LSE, CNN1DE and the ensemble model - CDE for the NK-Cell dataset along with their corresponding labels. A red cell indicates a negative class, a green cell indicates a positive class.

| Model | n = 256 | | | n = 512 | | | n = 1024 | | |
|---|---|---|---|---|---|---|---|---|---|
| | Precision | Recall | F1 | Precision | Recall | F1 | Precision | Recall | F1 |
| LSE-small **(ours)** | 93.3 ± 5.44 | 88.9 ± 0.00 | 91.0 ± 2.56 | 95.3 ± 5.80 | 84.4 ± 5.44 | 89.4 ± 4.43 | **97.8 ± 4.44** | 80.0 ± 8.32 | 87.6 ± 4.52 |
| CNN1DE-small **(ours)** | 93.1 ± 9.04 | 82.2 ± 5.44 | 87.2 ± 6.45 | **100 ± 0.00** | 82.2 ± 5.44 | 90.2 ± 3.24 | **97.8 ± 4.44** | 86.7 ± 8.32 | 91.6 ± 4.86 |
| LSE-medium **(ours)** | **95.8 ± 5.18** | **93.3 ± 5.44** | **94.4 ± 3.52** | 97.5 ± 5.00 | 84.4 ± 5.44 | 90.4 ± 4.79 | 95.0 ± 6.12 | 80.0 ± 4.44 | 86.8 ± 4.34 |
| CNN1DE-medium **(ours)** | 83.6 ± 5.98 | 80.0 ± 8.32 | 81.7 ± 6.97 | 82.8 ± 4.47 | 84.4 ± 5.44 | 83.5 ± 3.58 | 90.0 ± 5.00 | 80.0 ± 4.44 | 84.7 ± 4.71 |
| LSE-large **(ours)** | 95.5 ± 5.57 | 91.1 ± 8.32 | 93.1 ± 5.79 | **100 ± 0.00** | 86.7 ± 4.44 | 92.8 ± 2.65 | 97.5 ± 5.00 | 86.7 ± 8.32 | 91.6 ± 6.09 |
| CNN1DE-large **(ours)** | 83.9 ± 9.36 | 80.0 ± 8.32 | 81.8 ± 8.49 | **100 ± 0.00** | 71.1 ± 11.3 | 82.6 ± 7.68 | 91.6 ± 7.09 | 68.9 ± 13.0 | 77.9 ± 8.83 |
| CDE-medium **(ours)** | 93.1 ± 5.69 | 86.7 ± 4.44 | 89.7 ± 4.34 | 97.5 ± 5.00 | 86.7 ± 8.32 | 91.6 ± 6.09 | 93.3 ± 5.55 | 86.7 ± 8.32 | 89.5 ± 4.57 |
| CDE-best **(ours)** | 95.6 ± 5.44 | 91.1 ± 4.44 | 93.2 ± 4.13 | **100 ± 0.00** | **91.1 ± 4.44** | **95.3 ± 2.35** | **97.8 ± 4.44** | **93.3 ± 8.89** | **95.3 ± 5.80** |

**Table 9.** Results obtained for single models - LSE and CNN1DE and CytoDNNEnsemble model for AML dataset. Results marked in **bolded** are the best results obtained across all the models. CDE-best model consists of the best LSE and CNN1DE models. CDE-medium model consists of the LSE and CNN1DE models in medium sizes.

| Model | n = 1024 | | | n = 2048 | | | 4096 | | |
|---|---|---|---|---|---|---|---|---|---|
| | Precision | Recall | F1 | Precision | Recall | F1 | Precision | Recall | F1 |
| LSE-small **(ours)** | 74.0 ± 3.67 | 58.8 ± 7.20 | 65.4 ± 5.31 | 73.8 ± 3.91 | 63.5 ± 4.78 | 68.1 ± 2.77 | 75.2 ± 4.96 | 72.4 ± 4.78 | 73.7 ± 4.14 |
| CNN1DE-small **(ours)** | 58.7 ± 3.78 | 66.5 ± 16.3 | 61.7 ± 8.69 | 57.5 ± 1.85 | 61.2 ± 17.8 | 57.5 ± 10.6 | 55.6 ± 3.7 | 67.1 ± 19.1 | 59.5 ± 8.03 |
| LSE-medium **(ours)** | 77.0 ± 5.80 | 60.0 ± 7.35 | 67.2 ± 5.60 | 74.0 ± 4.04 | 61.8 ± 4.92 | 67.3 ± 3.81 | 74.0 ± 3.77 | 68.8 ± 6.06 | 71.3 ± 4.77 |
| CNN1DE-medium **(ours)** | 66.3 ± 3.71 | 71.8 ± 12.0 | 68.1 ± 4.44 | 65.3 ± 2.73 | 68.8 ± 18.1 | 65.4 ± 8.78 | 67.3 ± 5.96 | 55.9 ± 15.3 | 59.6 ± 6.84 |
| LSE-large **(ours)** | 75.1 ± 4.83 | 60.0 ± 4.78 | 66.6 ± 4.10 | 74.3 ± 6.83 | 63.5 ± 3.00 | 68.2 ± 1.78 | 73.8 ± 2.47 | 71.2 ± 2.20 | 72.5 ± 1.95 |
| CNN1DE-large **(ours)** | 77.4 ± 9.93 | 68.2 ± 12.8 | 71.2 ± 5.69 | 73.5 ± 4.65 | 71.2 ± 23.1 | 69.8 ± 14.2 | 66.0 ± 9.45 | **86.5 ± 9.95** | 73.7 ± 3.14 |
| CDE-medium **(ours)** | 74.0 ± 4.52 | 74.7 ± 8.84 | 74.1 ± 5.76 | 72.5 ± 7.28 | 73.5 ± 7.67 | 72.7 ± 4.80 | 69.7 ± 6.36 | 70.0 ± 6.81 | 69.5 ± 4.35 |
| CDE-best **(ours)** | **83.5 ± 2.77** | **80.6 ± 6.34** | **81.9 ± 4.39** | **83.6 ± 6.91** | **77.7 ± 1.44** | **80.4 ± 3.12** | **85.5 ± 4.79** | 73.5 ± 7.44 | **78.7 ± 4.36** |

**Table 10.** Results obtained for single models - LSE and CNN1DE and CytoDNNEnsemble model for HEUvsUE dataset. Results marked in **bolded** are the best results obtained across all the models. CDE-best model consists of the best LSE and CNN1DE models. CDE-medium model consists of the LSE and CNN1DE models in medium sizes.

| Model | n = 256 | | | n = 512 | | | n = 1024 | | |
|---|---|---|---|---|---|---|---|---|---|
| | Precision | Recall | F1 | Precision | Recall | F1 | Precision | Recall | F1 |
| LSE-small **(ours)** | 89.6 ± 3.01 | 70.8 ± 2.64 | 79.1 ± 2.13 | 92.1 ± 3.97 | 66.7 ± 2.64 | 77.3 ± 2.37 | 96.8 ± 4.10 | 73.3 ± 3.33 | 83.4 ± 3.07 |
| CNN1DE-small **(ours)** | 74.5 ± 10.2 | 80.8 ± 17.0 | 75.1 ± 5.20 | 75.8 ± 5.73 | 71.7 ± 18.9 | 71.7 ± 7.72 | 80.6 ± 16.2 | 47.5 ± 31.7 | 48.8 ± 25.8 |
| LSE-medium **(ours)** | **92.3 ± 2.54** | 69.2 ± 2.04 | 79.0 ± 1.81 | **92.5 ± 5.04** | 67.5 ± 3.12 | 77.9 ± 0.74 | **97.9 ± 2.58** | 74.2 ± 4.08 | 84.3 ± 2.75 |
| CNN1DE-medium **(ours)** | 79.6 ± 11.5 | 64.2 ± 22.5 | 66.8 ± 12.0 | 74.1 ± 5.96 | 76.7 ± 19.8 | 73.1 ± 9.52 | 74.6 ± 11.2 | 75.8 ± 22.3 | 71.4 ± 9.36 |
| LSE-large **(ours)** | 91.2 ± 2.64 | 69.2 ± 2.04 | 78.7 ± 2.14 | 91.5 ± 2.42 | 71.7 ± 4.86 | 80.3 ± 3.24 | **97.9 ± 2.58** | 73.3 ± 4.25 | 83.8 ± 2.69 |
| CNN1DE-large **(ours)** | 73.0 ± 3.08 | 80.0 ± 8.08 | 76.0 ± 3.72 | 78.1 ± 5.04 | 75.8 ± 10.3 | 76.5 ± 5.82 | 73.4 ± 3.74 | 74.2 ± 15.0 | 72.8 ± 7.61 |
| CDE-medium **(ours)** | 82.6 ± 8.54 | 82.5 ± 7.17 | 81.9 ± 2.66 | 82.5 ± 6.35 | 78.3 ± 6.67 | 79.9 ± 2.96 | 89.0 ± 9.76 | **83.3 ± 6.97** | **85.6 ± 5.32** |
| CDE-best **(ours)** | 79.0 ± 3.42 | **86.7 ± 4.86** | **82.5 ± 2.35** | 80.9 ± 1.64 | **85.0 ± 3.33** | **82.9 ± 2.28** | 86.6 ± 10.2 | 82.5 ± 4.86 | 84.1 ± 5.12 |

**Table 11.** Results obtained for single models - LSE and CNN1DE and CytoDNNEnsemble model for HVTN dataset. Results marked in **bolded** are the best results obtained across all the models. CDE-best model consists of the best LSE and CNN1DE models. CDE-medium model consists of the LSE and CNN1DE models in medium sizes.

| Model | n = 256 | | | n = 512 | | | n = 1024 | | |
|---|---|---|---|---|---|---|---|---|---|
| | Precision | Recall | F1 | Precision | Recall | F1 | Precision | Recall | F1 |
| LSE-small **(ours)** | 56.7 ± 8.17 | 90.0 ± 20.0 | 68.7 ± 11.1 | **66.7 ± 0.00** | **100 ± 0.00** | **80.0 ± 0.00** | 66.7 ± 0.00 | **100 ± 0.00** | 80.0 ± 0.00 |
| CNN1DE-small **(ours)** | 56.7 ± 13.3 | 90.0 ± 20.0 | 69.3 ± 15.55 | 63.3 ± 6.67 | **100 ± 0.00** | 77.3 ± 5.33 | 66.7 ± 0.00 | **100 ± 0.00** | 80.0 ± 0.00 |
| LSE-medium **(ours)** | 53.3 ± 12.5 | 90.0 ± 20.0 | 66.7 ± 14.6 | **66.7 ± 0.00** | **100 ± 0.00** | **80.0 ± 0.00** | 66.7 ± 0.00 | **100 ± 0.00** | 80.0 ± 0.00 |
| CNN1DE-medium **(ours)** | 56.7 ± 8.17 | **100.0 ± 0.00** | 72.0 ± 6.53 | 63.3 ± 6.67 | 90.0 ± 20.0 | 74.0 ± 12.0 | 63.3 ± 6.67 | **100 ± 0.00** | 77.3 ± 5.33 |
| LSE-large **(ours)** | 56.7 ± 13.3 | 90.0 ± 20.0 | 69.3 ± 15.6 | **66.7 ± 0.00** | **100 ± 0.00** | **80.0 ± 0.00** | **73.3 ± 13.3** | **100 ± 0.00** | **84.0 ± 8.00** |
| CNN1DE-large **(ours)** | 48.0 ± 4.00 | **100 ± 0.00** | 64.8 ± 3.81 | 54.7 ± 10.5 | **100 ± 0.00** | 70.1 ± 8.80 | 56.7 ± 8.17 | **100 ± 0.00** | 72.0 ± 6.53 |
| CDE-medium **(ours)** | **60.0 ± 8.17** | **100 ± 0.00** | **74.7 ± 6.53** | 63.3 ± 6.67 | **100 ± 0.00** | 77.3 ± 5.33 | 63.3 ± 6.67 | **100 ± 0.00** | 77.3 ± 5.33 |
| CDE-best **(ours)** | **60.0 ± 8.17** | **100 ± 0.00** | **74.7 ± 6.53** | 63.3 ± 6.67 | **100 ± 0.00** | 77.3 ± 5.33 | 66.7 ± 0.00 | **100 ± 0.00** | 80.0 ± 0.00 |

**Table 12.** Results obtained for single models - LSE and CNN1DE and CytoDNNEnsemble model for NK-Cell dataset. Results marked in **bolded** are the best results obtained across all the models. CDE-best model consists of the best LSE and CNN1DE models. CDE-medium model consists of the LSE and CNN1DE models in medium sizes.

## D Latent Space Extractor Model architecture details

Table 13 presents the hyperparameters of the LSE model architecture. The LSE autoencoder hyperparameters for the small, medium, large models are listed in Tables 14, 15, 16.

**Table 13.** Hyperparameters of the LSE Model architecture

| Module | Type | Hyperparameters |
|---|---|---|
| Feature Extractor | CNN Encoder | - |
| Classifier | Linear | out=16 |
| | LeakyReLU | - |
| | Linear | out=1 |
| | Sigmoid | - |

**Table 14.** Hyperparameters of the autoencoder-small architecture (encoder and decoder)

| Module | Type | Hyperparameters |
|---|---|---|
| Encoder | Conv1D | kernel=5, stride=2, padding=1 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| | Conv1D | kernel=3,5 stride=2, padding=1 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| | Flatten | - |
| | Linear | out=latent_dim |
| Decoder | Linear | in=latent_dim |
| | Unflatten | - |
| | ConvTranspose1D | kernel=3, stride=2, padding=1, out_padding=0 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| | ConvTranspose1D | kernel=5, stride=2, padding=1, out_padding=1 |

**Table 15.** Hyperparameters of the autoencoder-medium architecture (encoder and decoder)

| Module | Type | Hyperparameters |
|---|---|---|
| Encoder | Conv1D | kernel=5, stride=2, padding=1 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| | Conv1D | kernel=3, stride=2, padding=1 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| | Conv1D | kernel=3, stride=2, padding=1 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| | Flatten | - |
| | Linear | out=latent_dim |
| Decoder | Linear | in=latent_dim |
| | Unflatten | - |
| | ConvTranspose1D | kernel=3, stride=2, padding=1, out_padding=0 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| | ConvTranspose1D | kernel=3, stride=2, padding=1, out_padding=0 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| | ConvTranspose1D | kernel=5, stride=2, padding=1, out_padding=1 |

**Table 16.** Hyperparameters of the autoencoder-large architecture (encoder and decoder)

| Module | Type | Hyperparameters |
|---|---|---|
| Encoder | Conv1D | kernel=5, stride=2, padding=1 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| | Conv1D | kernel=3, stride=2, padding=1 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| | Conv1D | kernel=3, stride=2, padding=1 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| | Conv1D | kernel=3, stride=2, padding=1 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| | Flatten | - |
| | Linear | out=latent_dim |
| Decoder | Linear | in=latent_dim |
| | Unflatten | - |
| | ConvTranspose1D | kernel=3, stride=2, padding=1, out_padding=0 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| | ConvTranspose1D | kernel=3, stride=2, padding=1, out_padding=0 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| | ConvTranspose1D | kernel=3, stride=2, padding=1, out_padding=0 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| | ConvTranspose1D | kernel=5, stride=2, padding=1, out_padding=1 |

# E 1D Convolutional Neural Network Extractor Model architecture details

The hyperparameters of the CNN1DE model architecture are provided for all three size configurations: small (Table 17), medium (Table 18), and large (Table 19).

**Table 17.** Hyperparameters of the CNN1D Extractor small architecture

| Module | Type | Hyperparameters |
|---|---|---|
| Feature Extractor | Conv1D | kernel=5, stride=1, padding=1 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| | MaxPool1D | kernel=2, stride=2 |
| | Conv1D | kernel=3, stride=1, padding=1 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| Classifier | Linear | out=16 |
| | LeakyReLU | - |
| | Linear | out=1 |
| | Sigmoid | - |

**Table 18.** Hyperparameters of the CNN1D Extractor medium architecture

| Module | Type | Hyperparameters |
|---|---|---|
| Feature Extractor | Conv1D | kernel=5, stride=1, padding=1 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| | MaxPool1D | kernel=2, stride=2 |
| | Conv1D | kernel=3, stride=1, padding=1 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| | MaxPool1D | kernel=2, stride=2 |
| | Conv1D | kernel=3, stride=1, padding=1 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| Classifier | Linear | out=16 |
| | LeakyReLU | - |
| | Linear | out=1 |
| | Sigmoid | - |

**Table 19.** Hyperparameters of the CNN1D Extractor large architecture

| Module | Type | Hyperparameters |
|---|---|---|
| Feature Extractor | Conv1D | kernel=5, stride=1, padding=1 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| | MaxPool1D | kernel=2, stride=2 |
| | Conv1D | kernel=3, stride=1, padding=1 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| | MaxPool1D | kernel=2, stride=2 |
| | Conv1D | kernel=3, stride=1, padding=1 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| | MaxPool1D | kernel=2, stride=2 |
| | Conv1D | kernel=3, stride=1, padding=1 |
| | BatchNorm1D | - |
| | LeakyReLU | - |
| Classifier | Linear | out=16 |
| | LeakyReLU | - |
| | Linear | out=1 |
| | Sigmoid | - |

## F CytoDNNEnsemble Model architecture details

Table 20 presents hyperparameters of the final, CDE Model architecture.

**Table 20.** Hyperparameters of the CytoDNNEnsemble Model architecture

| Module | Type | Hyperparameters |
|---|---|---|
| Feature Extractor | Latent Space Extractor | - |
| | CNN1DE Extractor | - |
| Classifier | Linear | out=hidden_dim |
| | LeakyReLU | - |
| | Linear | out=1 |
| | Sigmoid | - |

## G Final hyperparameter values

Grid-search-optimized hyperparameter values for all models are listed in tables. Table 21 reports the values for the CDE model, Table 22 shows the values for the Latent Space Extractor model, Table 23 presents the selected values for the CNN1D Extractor model.

**Table 21.** Final Hyperparameter values of the CDE Models

| Dataset | Version | Hyperparameter | Value |
|---|---|---|---|
| AML | Medium | classifier hidden dim | 2 |
| | | learning rate | $10^{-2}$ |
| | | train feature extractors | true |
| | | weight decay | $10^{-2}$ |
| | Best | classifier hidden dim | 2 |
| | | learning rate | $10^{-1}$ |
| | | train feature extractors | false |
| | | weight decay | $10^{-1}$ |
| HEUvsUE | Medium | classifier hidden dim | 8 |
| | | learning rate | $10^{-2}$ |
| | | train feature extractors | true |
| | | weight decay | $10^{-4}$ |
| | Best | classifier hidden dim | 4 |
| | | learning rate | $10^{-2}$ |
| | | train feature extractors | true |
| | | weight decay | $10^{-3}$ |
| HVTN | Medium | classifier hidden dim | 2 |
| | | learning rate | $10^{-2}$ |
| | | train feature extractors | false |
| | | weight decay | $10^{-3}$ |
| | Best | classifier hidden dim | 2 |
| | | learning rate | $10^{-2}$ |
| | | train feature extractors | false |
| | | weight decay | $10^{-1}$ |
| NK-Cell | Medium | classifier hidden dim | 2 |
| | | learning rate | $10^{-2}$ |
| | | train feature extractors | false |
| | | weight decay | $10^{-3}$ |
| | Best | classifier hidden dim | 2 |
| | | learning rate | $10^{-3}$ |
| | | train feature extractors | true |
| | | weight decay | $10^{-2}$ |

**Table 22.** Final Hyperparameter values of the Latent Space Extractor Models

| Dataset | Size | Hyperparameter | Value |
|---|---|---|---|
| AML | Small | learning rate | $10^{-3}$ |
| | | learning rate autoencoder | $10^{-5}$ |
| | | latent dim autoencoder | 32 |
| | Medium | learning rate | $10^{-3}$ |
| | | learning rate autoencoder | $10^{-5}$ |
| | | latent dim autoencoder | 32 |
| | Large | learning rate | $10^{-3}$ |
| | | learning rate autoencoder | $10^{-6}$ |
| | | latent dim autoencoder | 64 |
| HEUvsUE | Small | learning rate | $10^{-3}$ |
| | | learning rate autoencoder | $10^{-6}$ |
| | | latent dim autoencoder | 32 |
| | Medium | learning rate | $10^{-4}$ |
| | | learning rate autoencoder | $10^{-6}$ |
| | | latent dim autoencoder | 64 |
| | Large | learning rate | $10^{-3}$ |
| | | learning rate autoencoder | $10^{-6}$ |
| | | latent dim autoencoder | 32 |
| HVTN | Small | learning rate | $10^{-3}$ |
| | | learning rate autoencoder | $10^{-5}$ |
| | | latent dim autoencoder | 64 |
| | Medium | learning rate | $10^{-3}$ |
| | | learning rate autoencoder | $10^{-5}$ |
| | | latent dim autoencoder | 32 |
| | Large | learning rate | $10^{-3}$ |
| | | learning rate autoencoder | $10^{-6}$ |
| | | latent dim autoencoder | 32 |
| NK-Cell | Small | learning rate | $10^{-3}$ |
| | | learning rate autoencoder | $10^{-6}$ |
| | | latent dim autoencoder | 32 |
| | Medium | learning rate | $10^{-3}$ |
| | | learning rate autoencoder | $10^{-5}$ |
| | | latent dim autoencoder | 32 |
| | Large | learning rate | $10^{-4}$ |
| | | learning rate autoencoder | $10^{-5}$ |
| | | latent dim autoencoder | 32 |

**Table 23.** Final Hyperparameter values of the CNN1DE Extractor Models

| Dataset | Size | Hyperparameter | Value |
|---|---|---|---|
| AML | Small | learning rate | $10^{-3}$ |
| | Medium | learning rate | $10^{-4}$ |
| | Large | learning rate | $10^{-2}$ |
| HEUvsUE | Small | learning rate | $10^{-1}$ |
| | Medium | learning rate | $10^{-2}$ |
| | Large | learning rate | $10^{-1}$ |
| HVTN | Small | learning rate | $10^{-4}$ |
| | Medium | learning rate | $10^{-3}$ |
| | Large | learning rate | $10^{-4}$ |
| NK-Cell | Small | learning rate | $10^{-2}$ |
| | Medium | learning rate | $10^{-2}$ |
| | Large | learning rate | $10^{-4}$ |

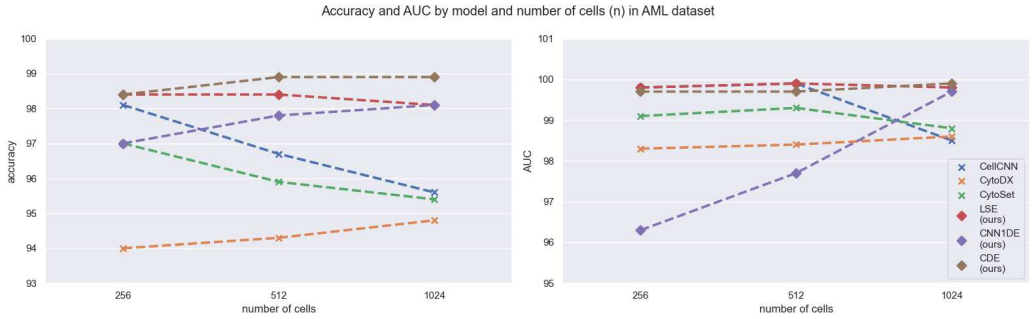



**Figure 10.** Accuracy and AUC metric values obtained for baseline models - CellCNN, CytoDX, CytoSet, our models - LSE, CNN1DE, CDE for AML dataset.

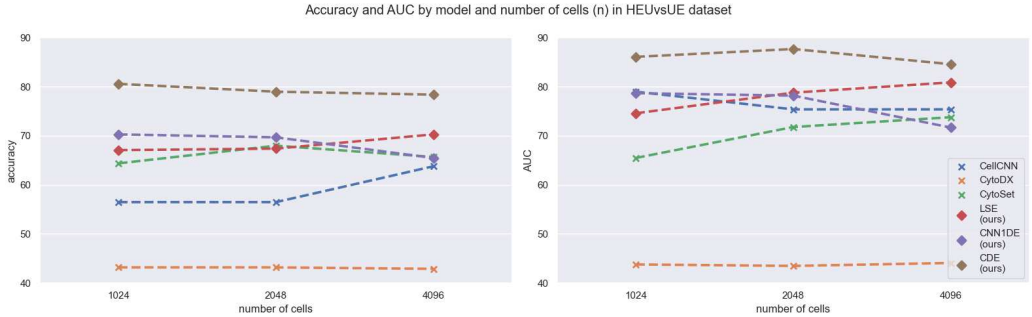



**Figure 11.** Accuracy and AUC metric values obtained for baseline models - CellCNN, CytoDX, CytoSet, our models - LSE, CNN1DE, CDE for HEUvsUE dataset.

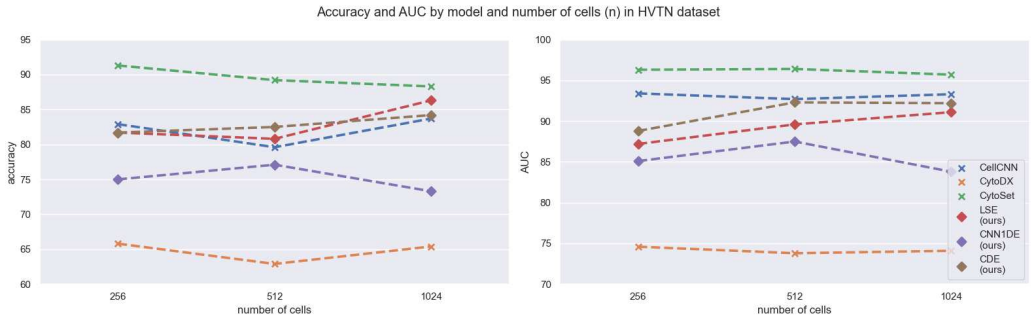



**Figure 12.** Accuracy and AUC metric values obtained for baseline models - CellCNN, CytoDX, CytoSet, our models - LSE, CNN1DE, CDE for HVTN dataset.

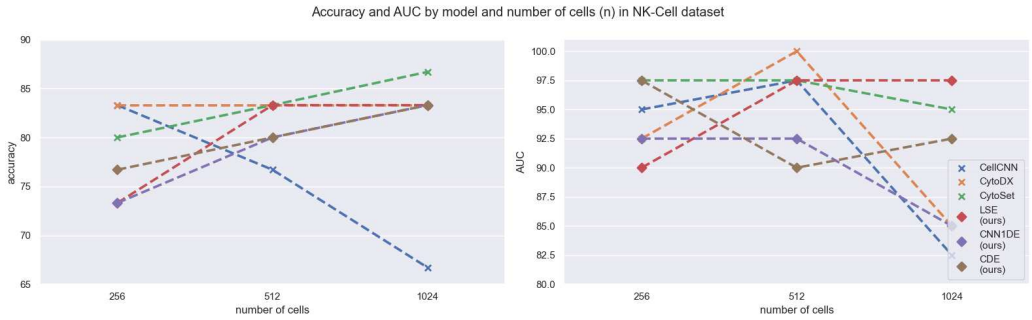



**Figure 13.** Accuracy and AUC metric values obtained for baseline models - CellCNN, CytoDX, CytoSet, our models - LSE, CNN1DE, CDE for NK-Cell dataset.

## H Result plots

The results for all three proposed models—CNN1DE, LSE, and CDE—alongside the baseline models CellCNN, CytoDX, and CytoSet are shown in Figures 10, 11, 12, and 13. In particular, on the larger AML and HEUvsUE datasets, the proposed ensemble model is largely agnostic to the sampling value $n$, consistently achieving the best performance across all sampling settings.

The figures also highlight the complementary behavior of the individual models in several cases. This effect is particularly pronounced at the largest sampling values, namely $n = 4096$ for HEUvsUE, $n = 1024$ for HVTN, and $n = 1024$ for NK-Cell. In these scenarios, the prediction quality of the CNN1DE model decreases, whereas that of the LSE model improves, resulting in more robust and general predictions across different input settings.