Supplementary Materials
NoTAC: A Noise-Tolerance Automatic Cleaning Framework for Bone Marrow Karyotyping Data

Rihan Huang*[1,2,3], Siyuan Chen*[1,2,3], Yafei Li[4], Chunling Zhang[5], Yilan Zhang[1,2,3], Changchun Yang[1,2,3], Na Li[6], Jingdong Hu[6], Ao Xu[6], Junkai Su[6], Xin Gao†[1,2,3], Huidan Li†[5], and Jiatao Lou†[5]

[1]Computer Science Program, Computer, Electrical and Mathematical Sciences and Engineering (CEMSE) Division, King Abdullah University of Science and Technology (KAUST), Thuwal, 23955-6900, Kingdom of Saudi Arabia
[2]Center of Excellence on Smart Health, King Abdullah University of Science and Technology (KAUST), Thuwal 23955-6900, Kingdom of Saudi Arabia.
[3]Center of Excellence for Generative AI, King Abdullah University of Science and Technology (KAUST), Thuwal 23955-6900, Kingdom of Saudi Arabia
[4]The First Affiliated Hospital of Zhengzhou University, Zhengzhou 450052, China
[5]Shanghai General Hospital, Shanghai 201613, China
[6]Smiltec (Suzhou) Co., Ltd., Room 401B, Building B6, No. 218 Xinghu Street, Suzhou Industrial Park, Suzhou, Jiangsu, China

Contents

*Contribute equally
†Corresponding Author. Email: xin.gao@kaust.edu.sa, huidan.li1@shgh.cn, loujiatao@sjtu.edu.cn

1 Detailed DNN Structures

Supplementary Table S1: Comparison of DNN structures: ResNet18, ResNet34, and VGG16

Layer	ResNet18 [1]	ResNet34 [1]	VGG16 [3]
Input	128 × 128 RGB image		
Initial	7×7, 64, stride 2 3×3 max pool, stride 2	7×7, 64, stride 2 3×3 max pool, stride 2	3×3, 64 (2 layers) 2×2 max pool
Conv2_x	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix} \times 3$	3×3, 128 (2 layers) 2×2 max pool
Conv3_x	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix} \times 4$	3×3, 256 (3 layers) 2×2 max pool
Conv4_x	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix} \times 6$	3×3, 512 (3 layers) 2×2 max pool
Conv5_x	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix} \times 3$	3×3, 512 (3 layers) 2×2 max pool
Output	Global average pool 24-d fc, softmax		4096-d fc 4096-d fc 24-d fc, softmax
Parameters	11.7M	21.8M	138M

Table S1 presents the detailed architectural specifications of the deep neural networks employed in our study: ResNet18 [1], ResNet34 [1], and VGG16 [3]. All three networks are designed for image classification with 128×128 RGB input images. The ResNet architectures (ResNet18 and ResNet34) implement residual learning through identity shortcut connections, which enable training of deeper networks by addressing the vanishing gradient problem. ResNet18 consists of 17 convolutional layers organized in four stages with [2,2,2,2] residual blocks, while ResNet34 extends this to 33 convolutional layers with [3,4,6,3] residual blocks per stage. Each residual block contains two 3×3 convolutional layers with batch normalization and ReLU activation.

In contrast, VGG16 follows a simpler uniform architecture without skip connections, comprising 13 convolutional layers and 3 fully connected layers. The VGG16 architecture uses small 3×3 convolutional filters throughout, with 2×2 max pooling operations to reduce spatial dimensions. The final classification layer in all networks outputs 24-dimensional class vectors, followed by softmax activation. Notable differences include the parameter efficiency of ResNet architectures (11.7M for ResNet18, 21.8M for ResNet34) compared to VGG16 (138M parameters), and the use of global average pooling in ResNets versus multiple fully connected layers in VGG16. These architectural choices reflect different design philosophies: ResNets prioritize depth and parameter efficiency through residual connections, while VGG16 demonstrates the effectiveness of uniform, deep convolutional architectures.

2 Detailed Analysis of Label Error Detection Performance in High-Noise Scenarios

Supplementary Table S2: NoTAC Detection Performance Metrics

Tasks	Performance Metric	Value
Label noise detection	Recall	0.9268
	Precision	0.7849
Image classification	Accuracy (Before Cleaning)	39.45%
	Accuracy (After Cleaning)	84.34%
	Relative Improvement	113.8%

To provide a comprehensive assessment of NoTAC's capability in identifying artificially introduced label errors, we conducted a detailed analysis of the framework's detection performance on the high-noise dataset. As described in Section IV-B, the artificially perturbed dataset contained approximately 50% overall noise rate, representing an extreme testing condition that exceeds typical clinical scenarios. The results of the NoTAC identification of artificially adulterated labels is shown in Table S2.

The achieved recall rate of 92.68% is particularly significant for clinical applications. In medical diagnostic contexts, it is crucial to identify the vast majority of problematic labels to maintain data quality and diagnostic reliability. Missing corrupted labels could lead to incorrect model training and potentially compromise diagnostic accuracy in clinical practice.

The relatively lower precision (78.49%) requires careful interpretation within our experimental design context. When constructing the artificially perturbed dataset, pre-existing natural label errors and outliers in the original clinical dataset were treated as "clean" samples for evaluation purposes. Consequently, when NoTAC correctly identifies these pre-existing issues as problematic, they are counted as false positives in our quantification metrics, artificially reducing the apparent precision.

This phenomenon suggests that NoTAC's actual performance may be underestimated in our evaluation, as the framework is detecting legitimate data quality issues that were not artificially introduced but existed naturally in the original dataset.

It is important to contextualize these results with real-world clinical scenarios, where label noise rates are typically much lower than our extreme 50% test condition. In our real-world case study (Section III-D-2), NoTAC demonstrated superior performance by successfully identifying all top 30 label errors in actual clinical data with natural noise patterns, achieving only one mis-prediction of the latent label. This finding provides strong evidence that NoTAC's detection performance is sufficiently robust and reliable for meaningful clinical application.

The combination of high recall (92.68%) and the demonstrated effectiveness in real clinical scenarios supports NoTAC's suitability for clinical adoption. The framework's ability to identify the vast majority of label errors while maintaining reasonable precision makes it a valuable tool for physicians and clinical researchers seeking to improve the quality of their karyotyping datasets.

3 Detailed Results of NoTAC

Table S3 presents a comprehensive comparison of model performance across four different data cleaning scenarios, providing detailed insights into the individual and combined effects of label noise and outlier removal on chromosome classification accuracy.

Baseline Performance ($\boldsymbol{X}$): The original dataset shows considerable performance variation across chromosome classes, with particularly challenging results for the sex chromosome Y (with recall: 0.3913 and precision: 0.2727). The overall accuracy of 0.8810 reflects the substantial impact of data quality issues on model performance.

Label Noise Removal Only ($\boldsymbol{X} \setminus \boldsymbol{X}_l$): Removing label errors while retaining outliers results in significant performance improvements across most chromosome classes, achieving an overall accuracy of 0.9361. Notably, chromosome Y recall improves dramatically from 0.3913 to 0.6250, demonstrating that label noise substantially impacts sex chromosome classification.

Outlier Removal Only ($\boldsymbol{X} \setminus \boldsymbol{X}_o$): Removing outliers while retaining label errors yields an overall accuracy of 0.9284. Although the recall of chromosome Y decreases to 0.4375 in this scenario compared to the label-noise-free condition, it is still better than the baseline (recall: 0.3913). These findings suggest that the removed outliers were indeed problematic samples rather than contributing model performance.

Complete NoTAC Framework ($\boldsymbol{X} \setminus (\boldsymbol{X}_l \cup \boldsymbol{X}_o)$): The combined removal of both label errors and outliers achieves the highest overall accuracy of 0.9399. Chromosome Y performance shows substantial improvement over the baseline: recall increases from 0.3913 to 0.6250, precision improves dramatically from 0.2727 to 0.8333, representing a more than three-fold improvement in precision while maintaining the enhanced recall achieved through label noise removal.

4 Hyperparameter Selection for NoTAC

The NoTAC framework employs two critical hyperparameters, K and f, which operate synergistically to distinguish between harmful outliers (technical artifacts) and beneficial outliers (rare but valid morphological variants).

Supplementary Table S3: Detailed results of NoTAC across data cleaning scenarios.

Chromosome	$\boldsymbol{X}$		$\boldsymbol{X} \setminus \boldsymbol{X}_l$		$\boldsymbol{X} \setminus \boldsymbol{X}_o$		$\boldsymbol{X} \setminus (\boldsymbol{X}_l \cup \boldsymbol{X}_o)$	
	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision
1	0.9540	0.9427	0.9971	0.9505	0.9971	0.9611	0.9914	0.9582
2	0.9560	0.9122	0.9855	0.9342	0.9827	0.9687	0.9769	0.9713
3	0.9540	0.9617	0.9728	0.9885	0.9546	0.9883	0.9773	0.9954
4	0.9360	0.8699	0.9637	0.9382	0.9728	0.8822	0.9486	0.9662
5	0.9020	0.9280	0.9381	0.9238	0.9567	0.9169	0.9474	0.9387
6	0.9320	0.8927	0.9840	0.9413	0.9707	0.9604	0.9867	0.9635
7	0.8720	0.8685	0.8872	0.9402	0.9098	0.9237	0.9248	0.9318
8	0.8720	0.9160	0.8859	0.9672	0.9069	0.9207	0.9399	0.9315
9	0.9360	0.9304	0.9736	0.9765	0.9560	0.9879	0.9765	0.9881
10	0.8400	0.8955	0.8831	0.9283	0.8929	0.9106	0.9773	0.8361
11	0.9460	0.9422	0.9673	0.9649	0.9849	0.9584	0.9874	0.9609
12	0.8920	0.9738	0.9716	0.9691	0.9612	0.9713	0.9302	0.9917
13	0.9260	0.9507	0.9540	0.9842	0.8972	0.9856	0.9234	0.9953
14	0.8820	0.8767	0.9260	0.9111	0.9315	0.8854	0.9041	0.9402
15	0.8880	0.7817	0.9337	0.8832	0.9367	0.8615	0.9578	0.8548
16	0.7080	0.9195	0.9468	0.9144	0.9539	0.8907	0.9610	0.9410
17	0.7680	0.8440	0.9326	0.7830	0.9476	0.7552	0.8876	0.7476
18	0.6980	0.8192	0.7500	0.9545	0.7597	0.9213	0.6916	0.9425
19	0.9180	0.7150	0.9197	0.9742	0.9246	0.9429	0.9708	0.9523
20	0.8960	0.8281	0.9444	0.9261	0.9167	0.9059	0.9762	0.9044
21	0.9000	0.8789	0.8381	0.9933	0.8239	0.9864	0.8977	0.9814
22	0.8520	0.8820	0.9793	0.8597	0.8905	0.8932	0.9290	0.9318
X	0.8580	0.8478	0.9595	0.8876	0.9150	0.9378	0.9231	0.9540
Y	0.3913	0.2727	0.6250	0.7692	0.4375	0.7778	0.6250	0.8333
Accuracy	0.8810		0.9361		0.9284		0.9399	

The careful selection of these parameters is essential for achieving optimal performance in chromosome image analysis.

4.1 Selection of the Neighborhood Parameter K

The choice of K in the k-nearest neighbors (KNN) algorithm presents a fundamental trade-off between computational efficiency and detection accuracy. The computational complexity scales directly with K, as the algorithm must compute and maintain distances to an increasing number of neighbors, thereby escalating both memory requirements and processing time.

Empirical analysis reveals that K values below 5 exhibit excessive sensitivity to local noise, failing to capture meaningful neighborhood structures within the feature space. Conversely, K values exceeding 20 impose substantial computational overhead without yielding proportional improvements in outlier detection accuracy. Through systematic evaluation on chromosome datasets, we determined that $K = 10$ provides an optimal balance between computational efficiency and detection performance, ensuring robust neighborhood estimation while maintaining tractable computational requirements.

4.2 Selection of the Threshold Parameter f

The parameter f governs the sensitivity of outlier detection through its role in OOD threshold calculation. This parameter directly influences the trade-off between false positive and false negative rates in outlier identification.

Values of f approaching 0 result in conservative outlier detection, characterized by elevated thresholds that preserve a larger proportion of samples. While this approach retains potential rare morphological variants, it simultaneously risks the retention of harmful technical artifacts that may compromise model performance. Conversely, larger f values establish more stringent detection thresholds, effectively removing obvious technical artifacts while carefully preserving samples that represent genuine structural variation.

Following the established methodology in prior work [2], we adopt $f = 0.37037$. This value has been empirically validated to achieve an optimal balance between the removal of clearly problematic samples and the retention

of morphologically diverse but valid chromosome variants, thereby preserving the biological diversity essential for robust model training.

5 The Zoomed Figures in the Manuscript

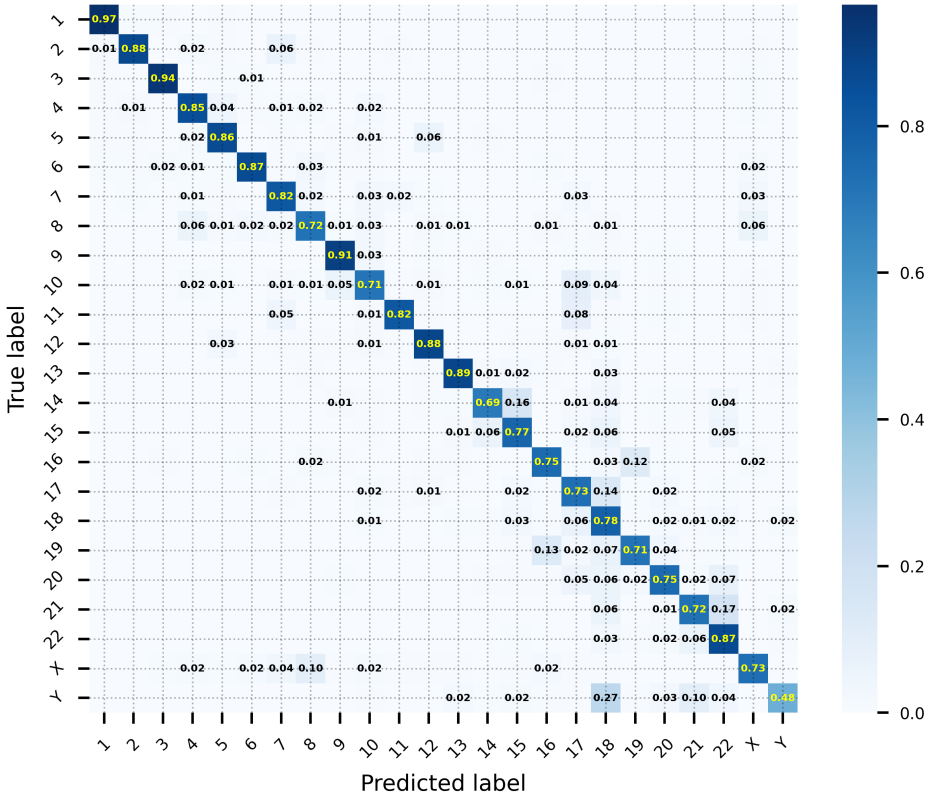
References

[1] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

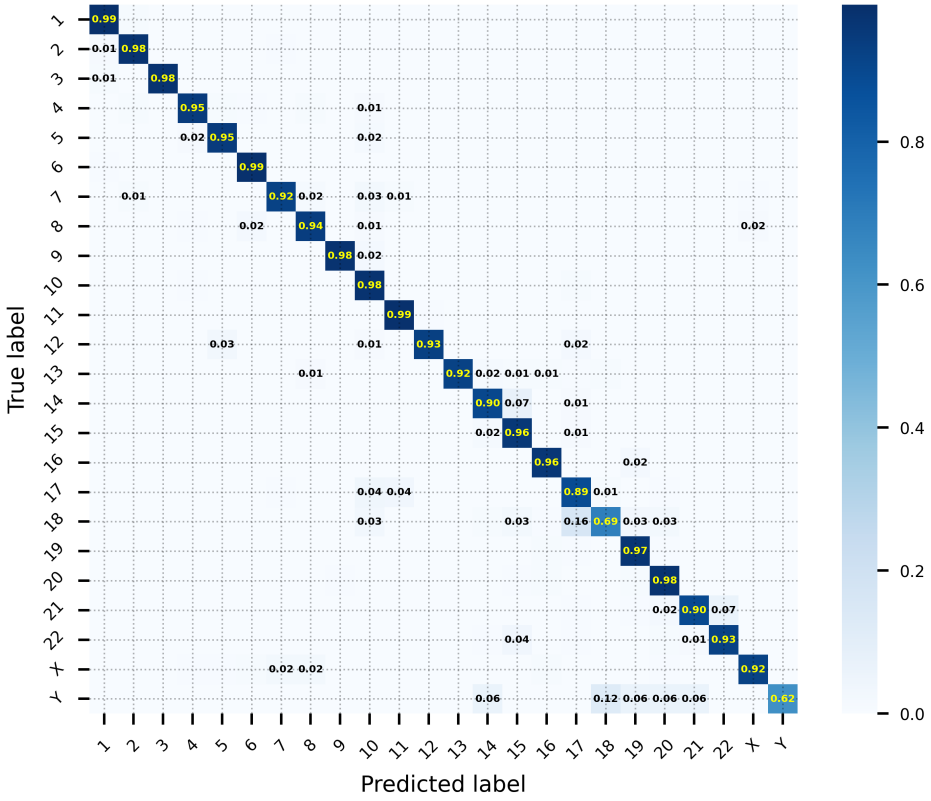
[2] J. Kuan and J. Mueller. Back to the basics: Revisiting out-of-distribution detection baselines. *arXiv preprint arXiv:2207.03061*, 2022.

[3] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

5.1 Confusion Matrices



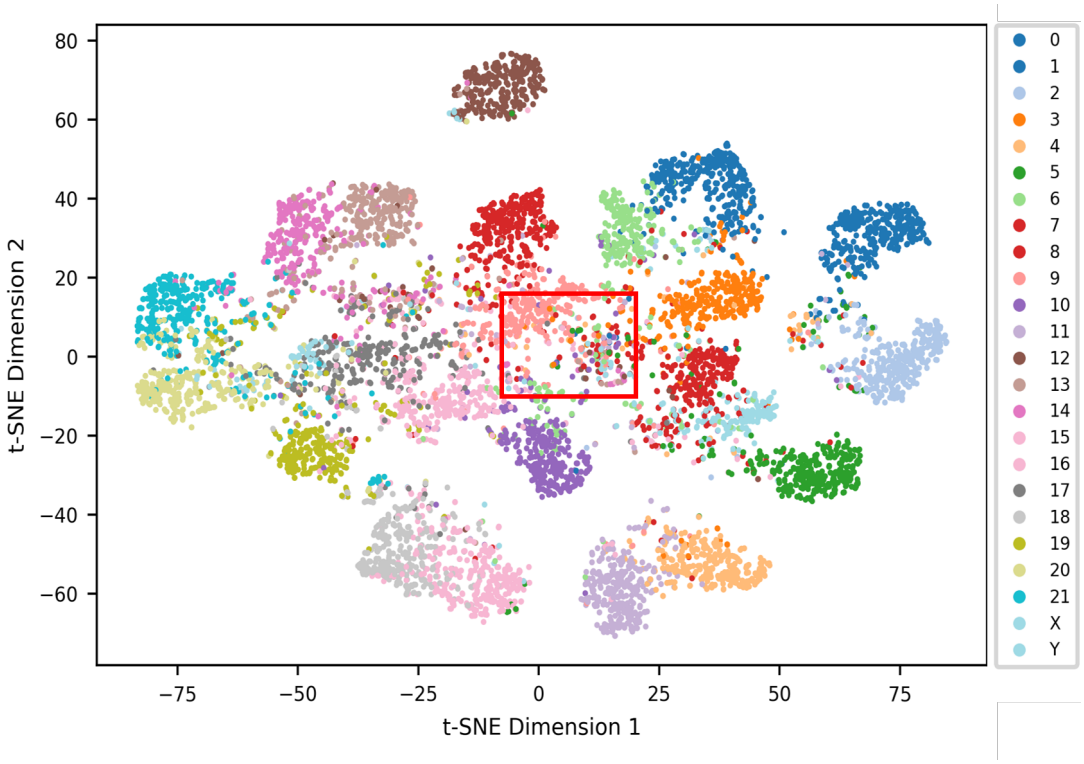
(a)



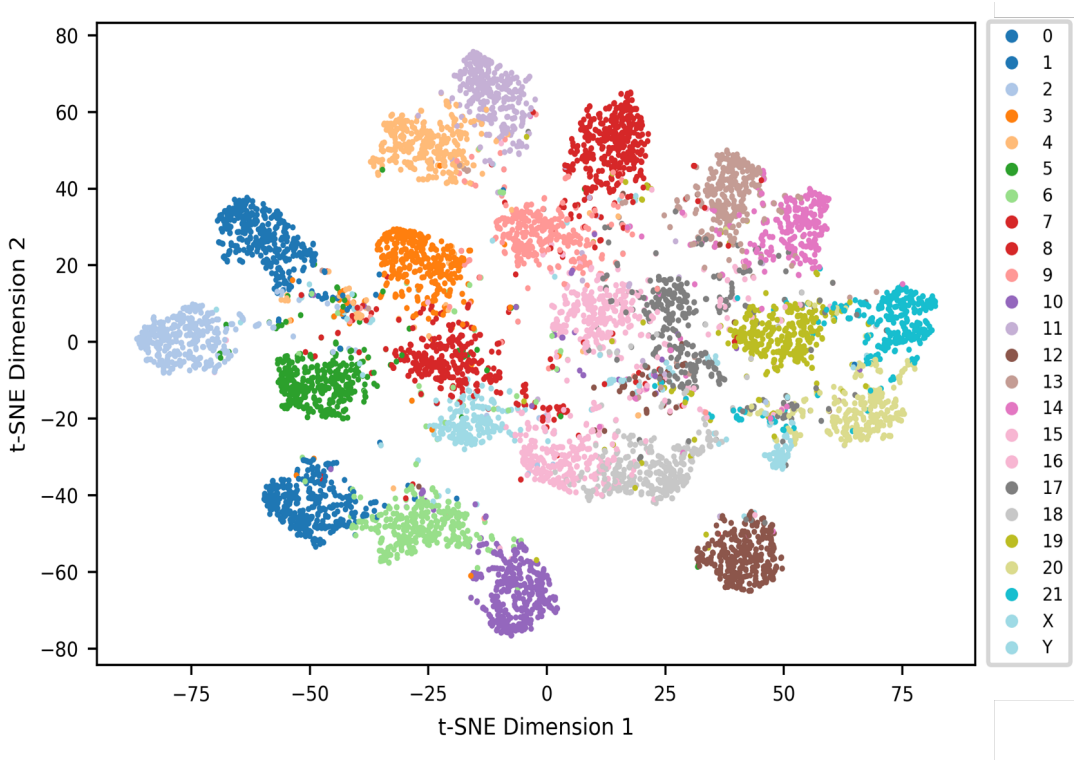
(b)

Supplementary Figure S1: Confusion matrices of the DNN model on (a) the original noisy data and (b) the refined data after NoTAC.

5.2 T-SNE Plots



(a)



(b)

Supplementary Figure S2: The t-SNE of the training features from (a) the original noisy data and (b) the refined data after applying NoTAC. The regions where features are mixed with each other are enclosed within a red box. Note that in the t-SNE plot, 8,000 examples were randomly sampled from the entire training dataset.