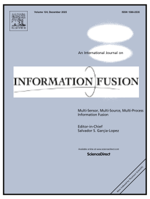
Information Fusion 130 (2026) 104088

Contents lists available at ScienceDirect

Information Fusion

journal homepage: www.elsevier.com/locate/inffus



# A comprehensive benchmark of spatial encoding methods for tabular data with deep neural networks

Jiayun Liu [a], Manuel Castillo-Cara [b,c,*], Raúl García-Castro [a]

[a] *Universidad Politécnica de Madrid, Madrid, Spain*
[b] *Universidad Nacional de Educación a Distancia, Madrid, Spain*
[c] *Instituto de Investigación Científica, Universidad de Lima, Lima, Peru*

ARTICLE INFO

*Keywords:*
Hybrid neural network
TINTOlib
Benchmark
Spatial encoding methods
Synthetic images
Tabular-to-image
Tabular2image

ABSTRACT

Despite the success of deep neural networks on perceptual data, their performance on tabular data remains limited, where traditional models still outperform them. A promising alternative is to transform tabular data into synthetic images, enabling the use of vision architectures such as Convolutional Neural Networks (CNNs) and Vision Transformers (ViTs). However, the literature lacks a large-scale, standardized benchmark evaluating these transformation techniques. This work presents the first comprehensive evaluation of nine spatial encoding methods across 24 diverse regression and classification datasets. We assess performance, scalability, and computational trade-offs under a unified framework with rigorous hyperparameter optimization. Our results reveal a performance landscape structured by data regimes, defined by sample size ($N$) and dimensionality ($d$), and show that the transformation method exerts a significantly stronger influence on predictive performance than the chosen vision architecture. In particular, REFINED emerges as the most robust transformation across tasks and datasets. Hybrid models (CNN + MLP, ViT + MLP) consistently reduce predictive variance, offering advantages especially in smaller datasets, yet play a secondary role. These findings suggest that transforming tabular data into synthetic images is a powerful, yet data-dependent, strategy. This benchmark provides clear guidance for researchers and practitioners, offering key insights into scalability, transformation behavior, and architectural interplay, establishing a comprehensive reference for future research on spatial encodings for tabular data.

## 1. Introduction

Deep learning models, such as Convolutional Neural Networks (CNNs) and Vision Transformers (ViTs), have demonstrated remarkable success in processing unstructured perceptual data such as images and text [1–3]. However, their application to tabular data, a format widely used in healthcare, finance and other domains [4,5], remains a significant challenge. The non-spatial, heterogeneous and permutation-invariant nature of tabular features [6] means that the strong inductive biases of vision models are not effective. Consequently, classical ensemble methods, particularly Gradient Boosted Decision Trees (GBDTs), often outperform DNNs [7,8], as their axis-aligned splitting mechanisms are well-suited to such data. This persistent performance gap is a critical and well-documented problem [6–8].

Recent efforts to bridge this gap involve the transformation of tabular data into synthetic images, allowing CNNs and ViTs to process tabular information [9,10]. This strategy, known as spatial encoding, proposes to re-engineer the data to fit the models. The central hypothesis is that by intelligently remapping order-free features onto a 2D grid, e.g., by placing similar features in close proximity [11,12], a meaningful spatial structure can be imposed. This allows the powerful inductive biases of vision models to extract complex patterns [13–15]. Various spatial encoding methods have been proposed, ranging from complex spatial encodings [11,12] and straightforward visualizations [16] to domain-specific applications [17,18].

Despite this promising activity, the field remains fragmented and a clear understanding of *when* and *why* these transformation methods work is lacking [6]. Comprehensive surveys [7] describe the landscape, but do not provide empirical performance comparisons. In contrast, individual method papers are often proofs-of-concept, demonstrating efficacy in specific domains [3,17,18], but lacking generalizability. Critically, there is no large-scale study that systematically compares these competing spatial encoding methods against each other and against strong classical baselines. Although hybrid CNNs have been explored [19], the systematic evaluation of modern Vision Transformers (ViT), both as standalone models and in hybrid (ViT + MLP) configurations, remains a significant and unaddressed area. This gap, often highlighted as a major hindrance to the progress of the field [5,6,8],

[*] Corresponding author.
*E-mail addresses:* jiayun.liu@upm.es (J. Liu), manuelcastillo@dia.uned.es (M. Castillo-Cara), r.garcia@upm.es (R. García-Castro).

https://doi.org/10.1016/j.inffus.2025.104088
Received 6 June 2025; Received in revised form 4 November 2025; Accepted 19 December 2025
Available online 25 December 2025

makes it difficult for practitioners to make informed, evidence-based decisions.

This paper directly addresses this research gap by presenting the first large-scale systematic benchmark of spatial encoding methods for tabular data. We evaluate nine representative transformation methods across 24 diverse regression and classification datasets, including the standardized OpenML-CC18 [20] and CTR23 [21] suites. To guide our investigation, we structure our systematic benchmark around the following key Research Questions (RQs):

**RQ1**: How does the predictive performance of spatial encoding methods compare against strong classical baselines across diverse data regimes (e.g., sample size, dimensionality, and feature type mix)?

**RQ2**: What are the computational trade-offs (e.g., transformation time, scalability) of different encoding families (parametric vs. non-parametric), and how do they impact feasibility for high-dimensional data?

**RQ3** To what extent do modern vision architectures (ViTs) and hybrid models (ViT + MLP, CNN + MLP), inspired by advances in multimodal learning [22–24], improve predictive performance or stability compared to standalone vision or classical models?

**RQ4** Which transformation methods and architectures are the most robust, and what are the key open challenges and insights (such as handling categorical features) for their practical application?

The remainder of this paper is organized as follows. Section 2 presents a structured literature review, categorizing transformation techniques into integrated, non-parametric, and parametric families, and motivating the need for a unified benchmark. Section 3 formalizes the selected spatial encoding methods and analyzes their algorithmic complexity. Section 4 introduces the experimental design, including datasets, evaluation protocols, and computational tools. Section 5 reports the empirical results across four experiments, comparing classical models, deep learning architectures, and hybrid networks, and includes statistical significance testing. Section 6 provides a qualitative and computational analysis of the generated synthetic images. Finally, Section 7 summarizes the main findings and outlines future research directions.

## 2. Related work

The literature on tabular data into synthetic image methods is diverse, but the methods can be critically analyzed along two primary axes: the transformation strategy (parametric vs. non-parametric) and the point of integration (preprocessing vs. integrated) [7,25].

### 2.1. Integrated (End-to-end) approaches

Unlike preprocessing-based methods, *integrated approaches* learn the transformation from tabular data to synthetic images jointly with the downstream task model. For example, HacNet [11] employs an attention-based generator to create image-like templates, which are co-optimized with the classifier. Tab2Vox [26] goes further, formulating the entire pipeline (including the transformation and CNN architecture) as a differentiable architecture search problem (DARTS) [27].

The theoretical promise of these methods is their ability to generate highly optimized, task-specific representations. However, this end-to-end optimization presents significant trade-offs that make them less suitable for a comparative benchmark: (1) they introduce significant computational complexity; (2) their modularity is reduced, as the transformation is not a reusable artifact separable from the model; and (3) the resulting black box layout is difficult to interpret and validate. For these reasons, our study focuses on the more modular, reproducible, and interpretable preprocessing-based approaches.

### 2.2. Preprocessing-based non-parametric methods

A second family of preprocessing methods consists of non-parametric techniques, which are often classified as "Image Marker" methods [25]. These methods use simple rule-based heuristics to map feature values directly onto an image grid without an underlying optimization. This category includes straightforward visualizations such as equidistant bar graphs (BarGraph) or Normalized Distance Matrix (DM) [28], as well as more abstract encodings. For instance, Super Tabular data Machine Learning (SuperTML) [29] renders feature values directly as text onto synthetic images, relying on the CNN to act as a character-level feature extractor. Other approaches like Vector-of-Feature Wrapping (FeatureWrap) [30] or Binary Image Encoding (BIE) [31,32] convert numerical values into their binary bit-patterns and arrange these bits into a synthetic image.

The clear advantage of this family is computational efficiency and simplicity. The methods are fast, deterministic, and easy to implement. However, they suffer from a critical conceptual limitation: the spatial layout is typically arbitrary (e.g., based on the original feature index) and is not designed to encode meaningful relationships between features. This limits the primary advantage of using vision models, which is their inherent ability to exploit spatial locality. Moreover, many of these methods scale poorly with feature dimensionality; a BarGraph, DM and Combination for high-dimensionality datasets becomes uninformative, and SuperTML suffers from unreadable, overlapping text.

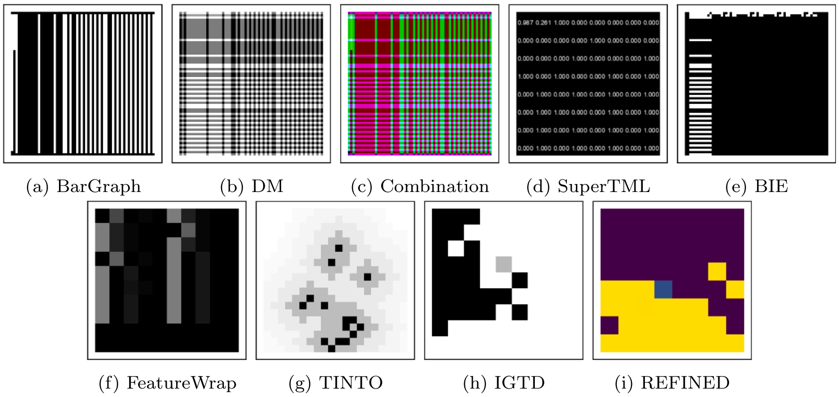
### 2.3. Preprocessing-based parametric methods

A significant family of methods computes the image layout as a fixed, data-driven preprocessing step. These methods are typically parametric as they rely on optimization algorithms to determine the spatial coordinates of each feature. Influential early work includes DeepInsight [13,33], which employs *t*-SNE or PCA to project high-dimensional features into a 2D space, and then rasterizes these positions onto synthetic images. REpresentation of Features as Images with NEighborhood Dependencies (REFINED) [14] builds on this by using Bayesian Metric Multidimensional Scaling (MDS) to create a layout that explicitly preserves the neighborhood dependencies between features. Similarly, Image Generator for Tabular Data (IGTD) [15] and its variants [3,18] optimize a feature-to-pixel mapping by minimizing the difference between feature similarity (in the original space) and pixel distance (on the 2D grid). Tabular data INTO synthetic image (TINTO) [17,34] also uses dimensionality reduction (PCA/*t*-SNE) but adds a controllable blurring technique, which helps to enlarge feature footprints and create smoother representations that are more amenable to convolutional filters.

The primary advantage of these parametric methods is their theoretical rigor; they generate spatially meaningful images where proximity is correlated with feature similarity. This makes them highly interpretable and well-suited for vision backbones. However, their reliance on complex optimizations such as *t*-SNE or MDS makes them computationally intensive, creating a significant scalability challenge that substantially increases the transformation time for datasets with thousands of features. Furthermore, the final layout can be sensitive to the hyperparameters of the reduction algorithm itself.

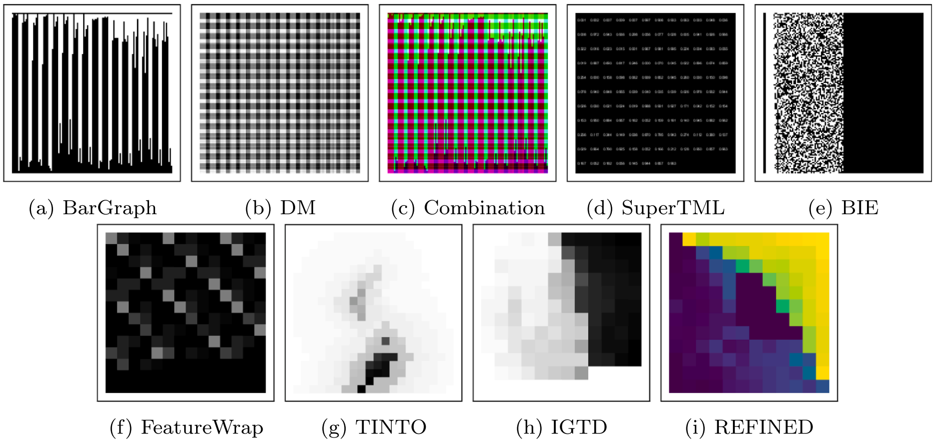
### 2.4. Benchmark rationale and method selection

Based on this critical analysis, our benchmark is designed to provide a clear and rigorous evaluation of the most reproducible and modular methods. Therefore, our study focuses on the Preprocessing-Based methods, as their modular nature permits a direct and standardized comparison of the spatial encoding artifacts themselves. We have selected nine representative techniques (formally defined in Section 3) drawn from the literature to span the two dominant competing families of this approach. This selection is deliberate: (1) the Parametric methods (TINTO, REFINED, IGTD) are included to test the central hypothesis that spatially meaningful, optimized layouts provide a tangible performance benefit; (2) the Non-Parametric methods (BarGraph, DM, Combination, SuperTML, FeatureWrap, BIE) are included to serve as computationally efficient, rule-based baselines. By systematically evaluating these two


(a) BarGraph (b) DM (c) Combination (d) SuperTML (e) BIE

(f) FeatureWrap (g) TINTO (h) IGTD (i) REFINED

**Fig. 1.** Visual representations of synthetic image methods applied to the Dengue dataset (Binary classification task).


(a) BarGraph (b) DM (c) Combination (d) SuperTML (e) BIE

(f) FeatureWrap (g) TINTO (h) IGTD (i) REFINED

**Fig. 2.** Visual representations of synthetic image methods applied to the GAS dataset (Multiclass classification task).

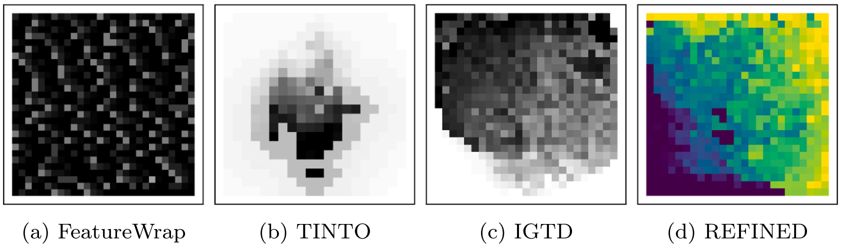

(a) FeatureWrap (b) TINTO (c) IGTD (d) REFINED

**Fig. 3.** Visual representations of synthetic image methods applied to the ISOLET dataset (Multiclass classification task).

distinct families against each other and against strong classical models, our benchmark provides a robust framework to rigorously address our research questions and map the practical trade-offs of these techniques.

## 3. A unified formalization of spatial encoding methods

Building on the landscape of approaches justified in Section 2, we now provide a formal description of the selected preprocessing-based methods. A key methodological contribution of this section is the introduction of a unified mathematical formalization for these techniques. In the literature, many transformation methods either lack a formal definition or are described with disparate notation, making a direct comparison difficult. Our unified framework, implemented in the open-source `TINTOlib` library [35,36], addresses this by ensuring that all methods are described in a consistent, shared formalism to clarify their core mechanics.

As justified in Section 2, our benchmark focuses on the two dominant families of preprocessing-based approaches. We selected nine representative methods: TINTO [17,34], IGTD [15] and REFINED [14] from the Parametric family; and BarGraph, DM, Combination [28], FeatureWrap [30], SuperTML [29], and BIE [32] from the Non-Parametric family.

*Notation.* Let $X \in \mathbb{R}^{N\times d}$ denote the dataset with $N$ instances and $d$ features. The $i$th row is $x_i \in \mathbb{R}^d$ and $x_{i,j}$ denotes its $j$th entry. Images are represented as matrices $M_i \in [0,255]^{h\times w\times c}$; when a single side is given, we use $w = h =$ `pixels`.

### 3.1. Non-parametric methods

Six non-parametric methods are provided: BarGraph, DM, Combination [28], SuperTML [29], FeatureWrap [30], and BIE [31,32]. These methods do not perform spatial optimization; instead, they translate features directly into visual representations.

#### 3.1.1. Equidistant bar graphs (BarGraph)

Equidistant bar graphs represent each instance as a vertical bar chart in which every feature is assigned a fixed column and the height of the bar is proportional to its normalized value $x'_{i,j}$. If $pixel\_width$ and $gap$ denote the width of the column and the inter-column gap in pixels, the width of the image $w$ is

$$w = d \cdot \text{pixel\_width} + (d+1) \cdot \text{gap}, \tag{1}$$

and the maximum drawable bar height (assuming a margin) equals

$$\text{bar\_height}_{\max} = h - 2 \cdot \text{pixel\_width}. \tag{2}$$

For instance $i$ and feature $j$ the height of the drawn bar is simply

$$\text{bar\_height}_{i,j} = \text{bar\_height}_{\max} \cdot x'_{i,j}. \tag{3}$$

This yields a one-channel image $M_i$ (see Figs. 1a and 2a).

### 3.1.2. Normalized distance matrix (DM)

DM encodes every pairwise difference of normalized features ($x'_{i,j}$) of the same instance into a square image of size $d \times d$. For an instance $i$ the matrix entries are given by

$$M_i[j,k] = |x'_{i,j} - x'_{i,k}|, \tag{4}$$

optionally scaled to a display range $[0, 255]$ when rendered (see Fig. 1b and 2b).

### 3.1.3. Combination of options (Combination)

The Combination method stacks three complementary encodings into a three-channel image. The first channel ($c = 1$) uses the DM representation, the second channel ($c = 2$) uses the BarGraph rendering, and the third ($c = 3$) contains a row-wise replication of the normalized feature vector $x'_i$. The final image is

$$M_i = \text{stack}(M_i^1,\ M_i^2,\ M_i^3), \tag{5}$$

where $M_i^1$, $M_i^2$, and $M_i^3$ correspond to these three components, respectively (see Figs. 1c and 2c).

### 3.1.4. Super tabular data machine learning (SuperTML)

SuperTML treats feature values as textual tokens placed on a square synthetic image. It has two variants. In the equal-font variant (SuperTML-EF), features are arranged in a regular grid with

$$n_{\text{cols}} = \left\lceil \sqrt{d} \right\rceil, \qquad n_{\text{rows}} = \left\lceil \frac{d}{n_{\text{cols}}} \right\rceil, \tag{6}$$

and each token occupies a cell of size $\text{cell\_w} = w/n_{\text{cols}}$ by $\text{cell\_h} = h/n_{\text{rows}}$. In the variable-font variant (SuperTML-VF), the size of the feature $j$ is scaled by an importance score $\text{imp}_j$ (for example, a Random Forest importance) so that

$$\text{font\_size}_j = \text{font\_size} \cdot \frac{\text{imp}_j}{\max_k \text{imp}_k}. \tag{7}$$

Text rendering produces a single-channel image depending on the chosen settings (see Figs. 1d and 2d).

### 3.1.5. Binary image encoding (BIE)

BIE [31,32] writes the IEEE-754 floating point bit pattern of each numerical feature such that each feature's bit pattern forms a column in the resulting image. Using precision $p \in \{32, 64\}$, the per-instance matrix $M_i$ has $p$ rows and $d$ columns, and each bit is mapped to black or white by

$$v_k = \begin{cases} 0 & b_k = 0, \\ 255 & b_k = 1, \end{cases} \tag{8}$$

where $b_k$ denotes the $k$th bit of the chosen floating-point representation (see Figs. 1e and 2e).

### 3.1.6. Vector-of-feature wrapping (FeatureWrap)

FeatureWrap converts features into a binary vector by applying one-hot encoding to categorical features and discretizing numerical features into $\text{bins}$ groups before one-hot encoding. If $c_j$ denotes the number of categories of the feature $j$ (or $\text{bins}$ for a discretized numerical feature), the total binary length is

$$L = \sum_{j=1}^{d} \begin{cases} c_j & \text{if feature } j \text{ is categorical}, \\ \text{bins} & \text{otherwise}. \end{cases} \tag{9}$$

The bit vector is padded to the required length $\text{values\_needed} = h \times w \times 8$, grouped into bytes, and each byte is converted to an integer value by

$$v = \sum_{b=0}^{7} b_b\, 2^{7-b}, \qquad b_b \in \{0, 1\}. \tag{10}$$

These integers populate the image grid, producing a one-channel image $M_i$ (see Figs. 1f, 2f, and 3a).

## 3.2. Parametric methods

Parametric transformation methods, i.e., TINTO [17,34], IGTD [15] and REFINED [14], optimize the spatial arrangement (or layout) of tabular data features using mathematical techniques (like dimensionality reduction or distance error minimization) to ensure that highly similar features are positioned in adjacent pixels, thus preserving neighborhood dependencies in the resultant 2D synthetic image representation.

### 3.2.1. Tabular data INTO synthetic images (TINTO)

TINTO [17,34] transforms tabular data into synthetic images by mapping each feature to a structured two-dimensional space. The process can be summarized in four steps: (1) Dimensionality reduction – project features to a 2D embedding; (2) Area delimitation – define and center a square region that contains all embedded points and scale it to the desired pixel resolution; (3) Coordinate calculation – map continuous coordinates to discrete pixel indices on the image grid; and (4) Blurring application – render each feature as a localized impulse and apply a radial halo to create spatial continuity and overlap smoothing.

For clarity, we keep the compact projection and mapping formulas: using PCA, the embedding is computed on the features (columns of $X$),

$$X_{\text{emb}} = (X^T - \mu)W, \quad W = [v_1, v_2], \tag{11}$$

where $X_{\text{emb}}$ is the $d \times 2$ matrix of the coordinates of the embedded features. The coordinates are then centered (using their mean $\mu_{\text{emb}}$) and scaled to the pixel grid as

$$\text{coord} = \frac{X_{\text{emb}} - \mu_{\text{emb}}}{R} \cdot (\text{pixels} - 1), \qquad R = \max_j \|X_{\text{emb},j} - \mu_{\text{emb}}\|_2. \tag{12}$$

Each mapped feature is rendered with a radial intensity profile; a convenient compact form for the halo is

$$I(r) = \min\left(\frac{\kappa}{\pi r^2}, 1\right), \tag{13}$$

for a scaling constant $\kappa$, and when multiple halos overlap their contributions are combined by a pointwise operator (e.g., maximum or mean). These steps produce a smooth spatial image $M_i$ per instance that preserves feature proximities from the embedding stage (see Figs. 1g, 2g and 3b).

### 3.2.2. Image generator for tabular data (IGTD)

IGTD [15] casts layout construction as an optimization over permutations: the goal is to order features on a grid so that similar features are placed close to each other. Practically the method follows three steps: (1) compute a feature similarity matrix $S$ using a chosen metric (e.g. Pearson, Spearman, or Euclidean on $X^T$); (2) build a candidate grid and compute its pairwise pixel distance matrix $D$ (with entries $D_{ij} = \|p_i - p_j\|_q$, $q \in \{1, 2\}$); and (3) search for a feature-to-pixel permutation $\pi$ that minimizes a discrepancy via greedy pairwise swaps until convergence.

**Table 1**
Algorithmic complexities for parametric and non-parametric methods implemented in TINTOlib and Deep Neural Network Models. $d$ is the number of features and $pixels$ denotes the dimensions (height and width) of the synthetic images.

| Type | Method | Complexity |
|---|---|---|
| Non-Parametric | BarGraph | $\mathcal{O}(N \times d)$ |
| | Combination | $\mathcal{O}(N \times d^2)$ |
| | DM | $\mathcal{O}(N \times d^2)$ |
| | SuperTML_EF | $\mathcal{O}(N \times d)$ |
| | SuperTML_VF | $\mathcal{O}(N \times d^2)$ |
| | BIE | $\mathcal{O}(N \times d)$ |
| | FeatureWrap | $\mathcal{O}(N \times d)$ |
| Parametric | TINTO (PCA) | $\mathcal{O}(N \times d^2 + d^3)$ |
| | TINTO ($t$-SNE) | $\mathcal{O}(N \times d^2)$ |
| | IGTD | $\mathcal{O}(N \times d^2)$ |
| | REFINED | $\mathcal{O}(N \times d^2 + d^3)$ |
| Model | MLP | $\mathcal{O}(d)$ |
| | CNN | $\mathcal{O}(\text{pixels}^2)$ |
| | ViT | $\mathcal{O}(\text{pixels}^2)$ |
| | MLP + CNN | $\mathcal{O}(d + \text{pixels}^2)$ |
| | MLP + ViT | $\mathcal{O}(d + \text{pixels}^2)$ |

Formally, if $S_{ij} = \text{sim}(X_{:,i}, X_{:,j})$ (similarity between the feature $i$ and $j$), the method seeks

$$\min_{\pi \in \mathcal{P}} \sum_{i,j} |S_{ij} - D^{\pi}_{ij}|^p, \qquad p \in \{1, 2\}, \tag{14}$$

where $D^{\pi}$ denotes $D$ after permuting the feature indices by $\pi$. The optimization is performed by iterative swaps that decrease the objective; the resulting permutation defines the final mapping from features to pixel coordinates used to construct $M_i$ (see Figs. 1h, 2h and 3c).

### 3.2.3. *REpresentation of features as images with NEighborhood dependencies (REFINED)*

REFINED [14] constructs layouts by first embedding the features using Multidimensional Scaling (MDS or Bayesian MDS) and then refining a discrete placement with a hill-climbing optimizer. The procedure is commonly described in four stages: (1) compute the feature distance matrix $D_f$ (typically Euclidean distances between feature vectors, $X_{:,i}$ and $X_{:,j}$); (2) apply MDS/BMDS to obtain a continuous 2D embedding; (3) snap each embedded point to the nearest unoccupied pixel in a predefined grid to obtain an initial discrete assignment; and (4) iteratively refine the assignment by local swaps (hill climbing) to better preserve original distances.

In compact form, the feature distance matrix is

$$D_{f,ij} = \|X_{:,i} - X_{:,j}\|_2, \tag{15}$$

the MDS coordinates ($mds$) are rank-normalized (used to break ties and place points) by

$$\text{mds\_norm}_j = \frac{\text{rank}(\text{mds}_j)}{d-1}, \tag{16}$$

and the refinement aims to find a permutation $\pi$ minimizing the placement discrepancy

$$\min_{\pi} \|D_f - D^{\pi}\|_F. \tag{17}$$

The final permutation $\pi$ defines the pixel positions for all features, ensuring that the correlated ones occupy the nearest pixels. Each instance $i$ is then rendered as an image $M_i$ (see Figs. 1i, 2i, and 3d).

### 3.3. *Algorithmic complexity*

Each of the implemented methods performs a different sequence of operations. Table 1 summarizes the temporal complexities in Big $\mathcal{O}$ notation for each of the algorithms implemented.

Among non-parametric methods, a clear distinction emerges. BarGraph, FeatureWrap, and SuperTML_EF are the most efficient, scaling linearly with the number of features ($\mathcal{O}(N \times d)$) due to their simple linear scan transformations. In contrast, Combination, DM, and SuperTML_VF scale quadratically with features ($\mathcal{O}(N \times d^2)$), a higher cost arising from operations dependent on feature pairs, such as computing the complete pairwise distance matrix or preliminary importance of the feature.

Parametric methods introduce a significant computational trade-off. Although some methods such as IGTD and TINTO (t-SNE) scale quadratically with features ($\mathcal{O}(N \times d^2)$), the most robust embedding methods (REFINED and TINTO (PCA)) introduce a cubic bottleneck ($\mathcal{O}(d^3)$). This cost, stemming from MDS optimization or eigendecomposition, makes their application computationally expensive for datasets with high dimensionality.

Finally, the DNN architectures also present a clear scaling distinction. MLPs scale linearly with the number of original features ($\mathcal{O}(d)$), while vision backbones (CNNs and ViTs) scale quadratically with image resolution ($\mathcal{O}(\text{pixels}^2)$). Hybrid models (e.g. MLP + CNN) simply combine these two costs additively ($\mathcal{O}(d + \text{pixels}^2)$).

This analysis highlights a fundamental trade-off: methods that create more robust spatial embeddings (e.g., REFINED) are often the most computationally expensive, particularly as feature dimensionality ($d$) increases. The practical implications of these complexities on measured image generation times are further explored in our qualitative analysis in Section 6.1. Next, we describe the benchmark setup used to systematically evaluate these transformation techniques across datasets and model architectures.

## 4. Experimental design and methodology

The section outlines principal tools, datasets, metrics, and guidelines used in the study.

### 4.1. *Tools and computational environment*

We conducted all model training and testing on the computing resources of the IPTC-AI. Innovation Space AI Supercomputing Cluster provided by the Universidad Politécnica de Madrid. The main features are 2 × Intel® Xeon® Gold 6240R (24 cores@2.4 GHz), 192 GiB RAM and one NVIDIA A100 GPU. The implementation is carried out in Python. We used PyTorch and Scikit-Learn libraries. Additionally, we used the TINTOlib library to transform tabular data into synthetic images [35,36].

### 4.2. *Benchmark datasets*

The benchmark covers both regression and classification tasks, including binary and multiclass problems[1]. These datasets span a wide range of sizes (from 506 to 581,012 instances), dimensionalities (8 to 1600 features) and feature types (purely numerical to mixed). This diversity allows evaluating how tabular structures and distributions affect image-based representations, as well as the scalability of transformation methods and models.

In addition to literature datasets, we include standardized OpenML benchmark suites, CC-18 [20] for classification and CTR23 [21] for regression, to ensure broad and reproducible coverage. Following a unified protocol, all datasets are stratified by size and feature composition using tertiles of $\log_{10}(N)$ (computed jointly across both suites) to define small, medium, and large bins, and a categorical-feature fraction threshold ($\geq 0.6$) to distinguish high-numerical and high-categorical regimes.

[1] **Binary:** QSAR, Bioresponse, HELOC, Nomao, Credit, Sick, Dengue, Adult. **Multiclass:** CNAE-9, GAS, ISOLET, CMC, DNA, Covertype, Connect. **Regression:** Music, Boston, Puma, California, Conductivity, MIMO, Student, Health.

**Table 2**
Datasets used in the benchmark, grouped by task type and characterized by size, feature composition, and dimensionality. “Num.-dom.”and “Cat.-dom.” denote numerical- and categorical-dominant datasets, respectively.

| Task | Dataset | #Samples | #Features | #Num. | #Cat. | Classes | Size Bin | Feature Comp. | Dimensionality |
|---|---|---|---|---|---|---|---|---|---|
| **Binary** | QSAR | 1055 | 41 | 41 | 0 | 2 | Small | Num.-dom. | Medium |
| | Bioresponse | 3751 | 1776 | 1776 | 0 | 2 | Medium | Num.-dom. | High |
| | HELOC | 9871 | 23 | 23 | 0 | 2 | Medium | Num.-dom. | Low |
| | Nomao | 34,465 | 118 | 89 | 29 | 2 | Large | Num.-dom. | Medium |
| | Credit | 690 | 15 | 6 | 9 | 2 | Small | Cat.-dom. | Low |
| | Sick | 3772 | 29 | 7 | 22 | 2 | Medium | Cat.-dom. | Low |
| | Dengue | 11,448 | 26 | 2 | 24 | 2 | Medium | Cat.-dom. | Low |
| | Adult | 48,842 | 14 | 6 | 8 | 2 | Large | Cat.-dom. | Low |
| **Multiclass** | CNAE-9 | 1080 | 856 | 856 | 0 | 9 | Small | Num.-dom. | High |
| | GAS | 13,910 | 128 | 128 | 0 | 6 | Medium | Num.-dom. | Medium |
| | ISOLET | 7797 | 617 | 617 | 0 | 26 | Medium | Num.-dom. | High |
| | CMC | 1473 | 9 | 2 | 7 | 3 | Small | Cat.-dom. | Low |
| | DNA | 3186 | 180 | 0 | 180 | 3 | Medium | Cat.-dom. | Medium |
| | Covertype | 581,012 | 54 | 10 | 44 | 7 | Large | Cat.-dom. | Medium |
| | Connect | 67,557 | 42 | 0 | 42 | 3 | Large | Cat.-dom. | Medium |
| **Regression** | Music | 1059 | 116 | 116 | 0 | — | Small | Num.-dom. | Medium |
| | Boston | 506 | 13 | 11 | 2 | — | Small | Num.-dom. | Low |
| | Puma | 8192 | 32 | 32 | 0 | — | Medium | Num.-dom. | Medium |
| | California | 20,640 | 8 | 8 | 0 | — | Large | Num.-dom. | Low |
| | Conductivity | 21,263 | 81 | 81 | 0 | — | Large | Num.-dom. | Medium |
| | MIMO | 252,004 | 1600 | 1600 | 0 | — | Large | Num.-dom. | High |
| | Student | 649 | 31 | 13 | 17 | — | Small | Cat.-dom. | Medium |
| | Health | 22,272 | 11 | 4 | 7 | — | Large | Cat.-dom. | Low |

This design enables systematic comparisons of tabular data into synthetic images methods and vision-based architectures under controlled capacity and training protocols. A complete summary of the data set, including task type, sample size, and feature composition, is reported in Table 2.

No additional preprocessing was applied to the data consumed by the transformation methods, except for the HELOC dataset, where rows containing only −9 values (treated as `NaN`) were removed. This preprocessing strategy enables a fair comparison of the inherent capabilities of each transformation method without introducing external biases.

### *4.3. Experimental setup*

*Models and architectures.* Our benchmark compares three model families: classical gradient-boosting models (XGBoost, CatBoost, LightGBM); deep neural networks (a standard MLP, a ResNet-style CNN, and a Vision Transformer); and hybrid architectures (CNN + MLP and ViT + MLP). The hybrid models process the generated image with a vision backbone and the raw tabular data with a parallel MLP branch, fusing the representations via concatenation (see Fig. 4).

*Data partitioning and preprocessing.* To ensure a consistent and reproducible evaluation, we employed a standardized preprocessing pipeline. Each data set was split into training sets (75%), validation (15%), and test (15%) using a fixed random seed; for all classification tasks, this split was stratified to preserve class distribution. Prior to encoding, missing values were handled: numerical features were imputed with the mean (or median for skewed distributions), while missing categorical values were treated as a distinct `__null__` category. Subsequently, all categorical features were converted to a numerical format using one-hot encoding, and all numerical features were scaled to a range $[0, 1]$ using min-max. To ensure compatibility with all vision architectures, particularly the fixed patch grid of ViT, images were processed to a uniform target dimension when the image size was not divisible and too large, thus reducing the number of tokens and maintaining computational efficiency. This was achieved by padding smaller images on the right and bottom with a constant value (the mean color of the training set), ensuring that the image dimensions were divisible by the ViT patch size without distorting the original content.

*Training and hyperparameter optimization.* We employed a unified, multi-stage hyperparameter optimization protocol for all models using Optuna. First, an initial search of 100 trials was conducted, with each trial trained for 50 epochs. To manage computational cost on large datasets ($N > 20k$), this search phase was performed on a 25% random subset of the training data. Following this, the top five configurations from the initial search were retrained for a full 100 epochs on the complete training set. The single best-performing configuration was selected based on task-specific validation metrics: Root Mean Squared Error (RMSE) for regression, ROC-AUC for binary classification, and Accuracy for multiclass classification. The search space covered core hyperparameters including learning rate, weight decay, learning rate schedule parameters (`div_factor`, `final_div_factor`, and `pct_start`), dropout, number of layers and hidden dimensions.

For the ViT models, we optimized the embedding dimension, depth, number of attention heads, and MLP expansion ratio to balance representational capacity with computational cost. CNNs were tuned over stem type, number of stages, block configuration, and base width. To ensure fair comparison, their total number of parameters was constrained to remain within 25% of the ViT counterpart, i.e., CNN trials exceeding this threshold were automatically pruned. In the hybrid models (ViT + MLP and CNN + MLP), we fixed the best-performing vision backbone from the standalone configuration and tuned only the fusion module, consisting of hidden dimensions and dropout. This setup ensures that all architectures are compared under equivalent training budgets and model capacities, isolating the effect of architectural design. All deep models used the `AdamW` optimizer with a `OneCycle` learning rate schedule. Batch size was adapted to dataset size: 16 for $N \leq 1k$, 32 for $1k < N \leq 5k$, 64 for $5k < N \leq 20k$, and 256 for $N > 20k$.

*Evaluation protocol.* The final selected model for each combination was retrained and evaluated five times with different random seeds on the held-out test set. All results are reported as the mean and standard deviation. For clarity in our main analysis, we report normalized RMSE for regression and accuracy for classification. A comprehensive and interactive breakdown of results for all performance and efficiency metrics across all experiments is publicly available on our project website[2].

[2] https://oeg-upm.github.io/TINTOlib/

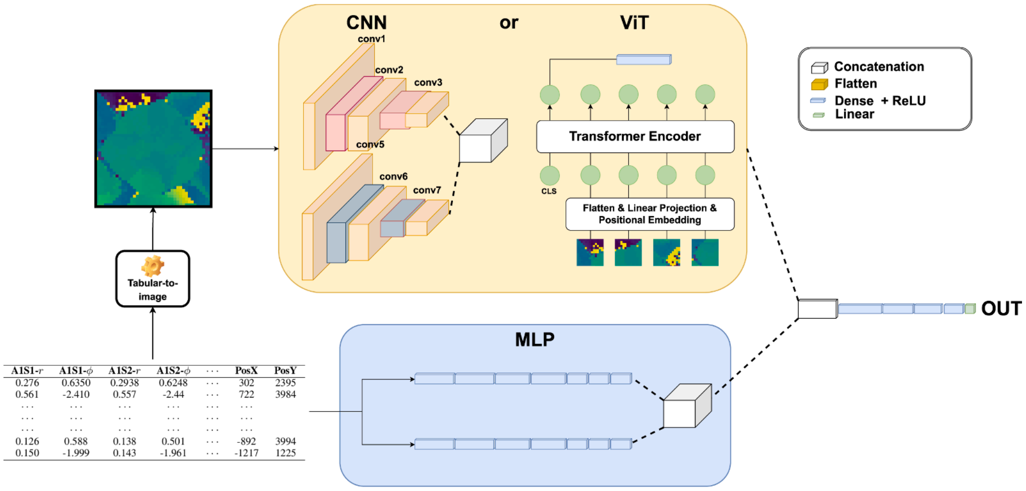


**Fig. 4.** A simplified illustration of the HyNN architecture. This figure shows that either CNN or ViT can be used as the vision component, while the MLP component processes tabular data directly. It also highlights the flexibility of the model, which allows for using only the vision or MLP part, illustrating the different architectures evaluated in this study.

## 5. Benchmark evaluation

This section presents a systematic evaluation of different machine learning approaches and transformation techniques for tabular data into synthetic images. We analyze the performance of classical models and DNNs, i.e., CNNs, ViTs, CNN + MLP and ViT + MLP, across multiple datasets. The experiments compare the effectiveness of these models in regression and classification tasks, highlighting the impact of various transformation methods. In addition, a statistical analysis is conducted to determine the significance of the performance differences between models and methods. This benchmark provides a comprehensive assessment of the advantages and limitations of each approach, offering insights into their applicability for tabular data analysis.

### *5.1. Global performance: Classical vs. neural pipelines*

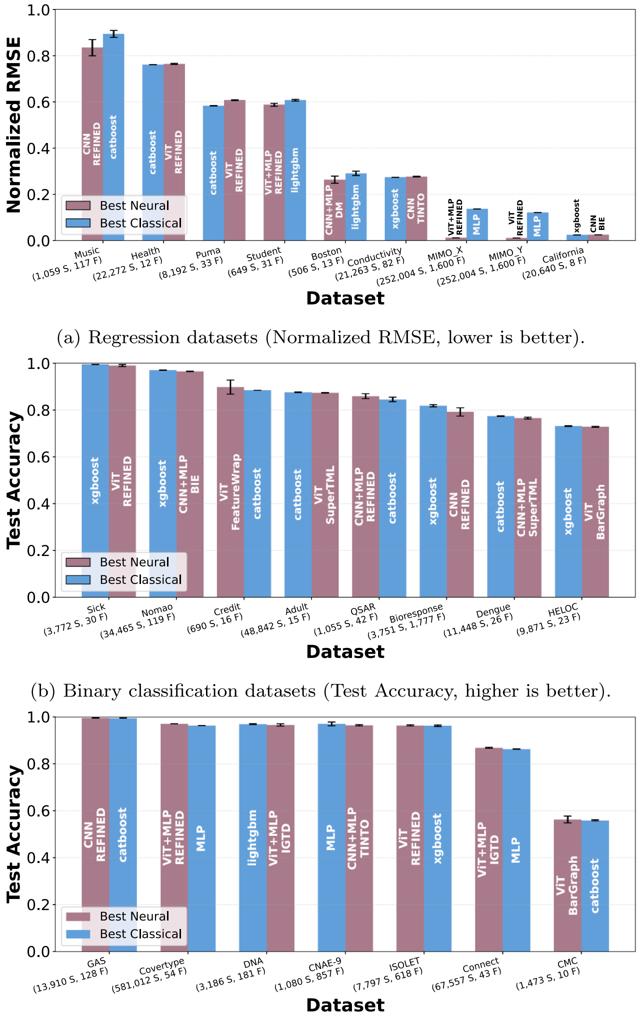
To address **RQ1** (see Section 1), we first conduct a head-to-head comparison of the best-performing classical model against the best-performing neural pipeline (i.e., the best combination of a spatial encoding method and a DNN architecture). Fig. 5 synthesizes this comparison across all regression, binary, and multiclass tasks.

The results in Fig. 5 reveal a clear and consistent pattern that is not governed by the type of task (e.g., regression vs. classification) but rather by the data regime, defined by the interplay of sample size ($N$) and feature dimensionality ($d$).

Neural/Image-based pipelines demonstrate a distinct advantage in two specific regimes:

- Data-Scarce Regimes ($N \lesssim 5k$): Neural models win consistently when samples are limited, regardless of task. This is evident in regression (e.g., Boston, Student), binary classification (Credit, QSAR), and multiclass classification (CMC).
- Extreme-Dimensionality Regimes ($d \gtrsim 1k$): Neural models dominate in high-dimensional settings, even when sample sizes are large. This is seen in regression (MIMO-X and MIMO-Y) and multiclass classification (ISOLET).

Classical models (predominantly GBDTs) are the dominant choice in data-rich (medium-to-large $N$) regimes where feature counts are low-to-moderate. This pattern holds true across regression (e.g., Health, California), binary classification (e.g., Sick, Nomao, Adult), and multiclass classification (e.g., DNA). Interestingly, classical models also demonstrate strong performance in some cases that fall outside this optimal zone,



(a) Regression datasets (Normalized RMSE, lower is better).

(b) Binary classification datasets (Test Accuracy, higher is better).

(c) Multiclass classification datasets (Test Accuracy, higher is better).

**Fig. 5.** Comparison between the best classical and neural models across (a) regression, (b) binary, and (c) multiclass datasets. Bars show mean performance across seeds with error bars for variability. Dataset labels denote samples (S) and features (F).

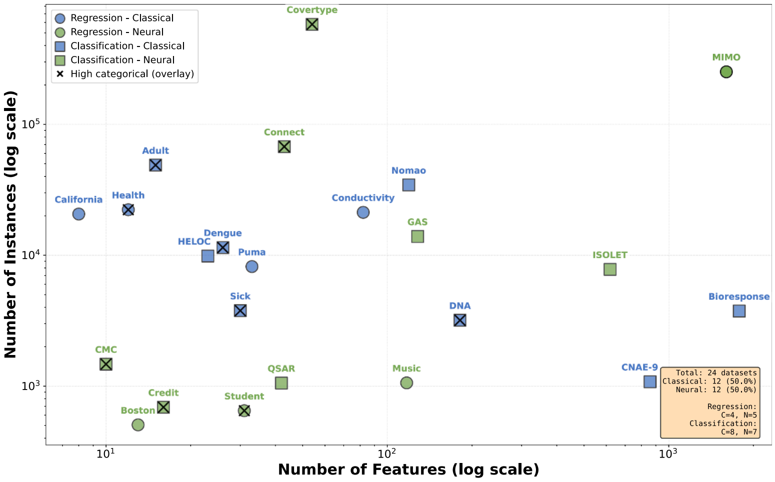


**Fig. 6.** Performance map across all problem types. Each point is a dataset plotted by number of features (x-axis, log scale) and number of instances (y-axis, log scale). Shape encodes problem type (circles = regression; squares = classification), color indicates whether a classical or a neural model achieved the best score, and an 'X' overlay marks datasets with a high proportion of categorical features. The plot highlights where classical vs. neural families tend to excel under different dataset regimes.

such as the high-dimensional Bioresponse dataset and the data-scarce, high-dimensional CNAE-9 dataset, highlighting their robustness.

This primary finding, that neural/image methods excel in low-$N$ or high-$d$ scenarios while classical models are superior for high-$N$, low-$d$ data, is the key answer to **RQ1**. Among the neural winners, the parametric method REFINED is the most frequent and reliable transformation, with other methods appearing in specific contexts.

### *5.2. Performance map and data regimes*

To synthesize the results for all tasks and provide a unified view of these regime effects, we introduce the performance map in Fig. 6. The figure places each data set in the $(\log_{10} d, \log_{10} N)$ plane, abstracting architectural details to highlight how the sample size, dimensionality, and feature composition relate to which family, classical tabular vs. neural image, based-achieves the best mean test score.

The map reveals three primary insights. First, it confirms there is no single overall winner; across all 24 benchmarks, classical and neural methods are in a statistical dead heat, with each family winning on exactly 12 datasets (50.0%). This reinforces that performance is driven by the characteristics of the dataset, not the type of task.

Second, the map visually synthesizes the data regime patterns identified in Section 5.2. Neural/image-based pipelines consistently dominate in the small-sample ($N \lesssim 5k$) and they also show a distinct advantage in specific high-dimensional cases, such as the extreme-dimensionality MIMO dataset and the high-dimensional ISOLET dataset. Conversely, classical models are the superior choice in the dense cluster of medium-to-large-$N$ problems with low-to-moderate feature counts.

Third, the map allows for an analysis of the composition of features. Datasets with a high proportion of categorical features (marked with an 'X') are distributed across various regimes and show no clear preference for one model family. This suggests that the presence of many categorical features is a secondary factor, and the primary driver of performance remains the interplay between sample size and dimensionality.

This unified map demonstrates that the performance landscape is a spectrum rather than a dichotomy. Classical and neural models excel in complementary regions of the $(N, d)$ space, and the boundary between them is modulated by the choice of transformation method and architectural enhancements such as hybrid designs. This integrated view sets the stage for the next section, where we test whether these observed patterns hold under formal significance analysis.

### *5.3. Analysis of architectures and transformation methods*

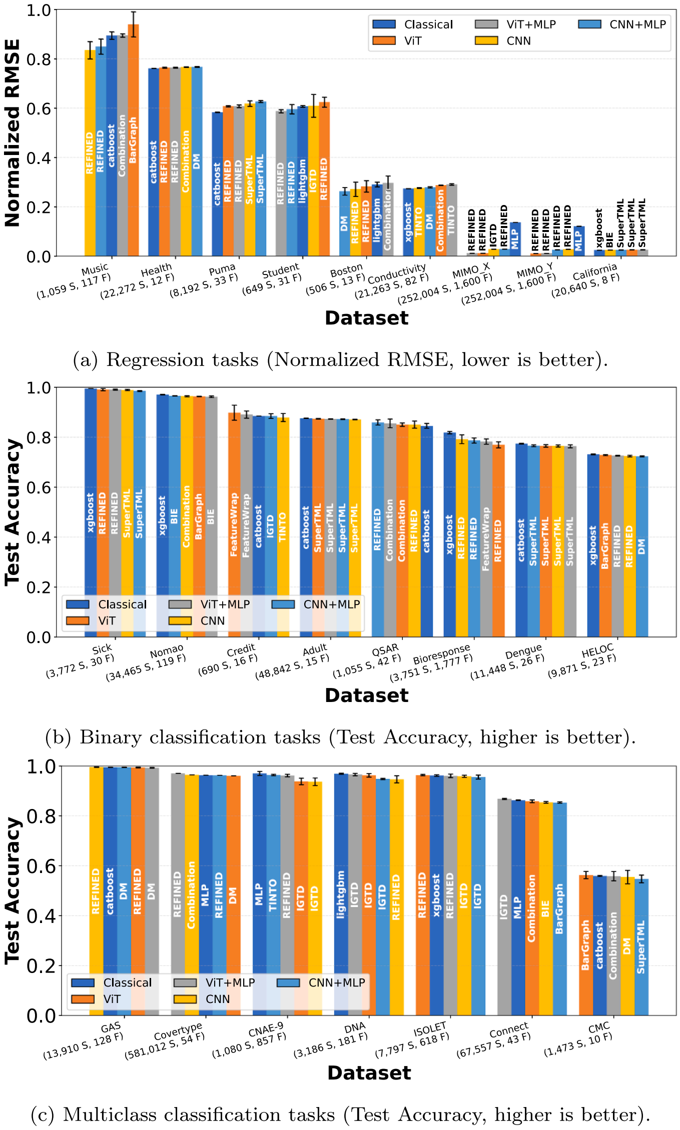
Having established the performance regimes relative to classical models, we now analyze the performance within the neural pipelines to address **RQ3** (impact of architectures) and **RQ4** (robustness of transformation methods). Fig. 7 presents the best-performing transformation method for each of the five neural architecture families (Classical, ViT, ViT + MLP, CNN, CNN + MLP) across all datasets.

*Architectural effects (**RQ3**).* A clear and consistent pattern emerges regarding hybrid architectures: hybrid models (ViT + MLP, CNN + MLP) consistently reduce predictive variance. This effect is visually evident in Fig. 7, where the error bars for hybrid models (e.g., grey and blue bars) are often narrower than their vision-only counterparts (orange and red bars). This variance reduction is most pronounced in small-sample datasets, e.g., Student, Boston, and QSAR. While vision-only models often achieve comparable or slightly better mean performance (especially in binary and multiclass tasks), the stabilization provided by the parallel MLP branch confirms that hybrid models are a more stable and reliable choice, particularly when data is scarce.

*Transformation method performance (**RQ4**).* Regarding the transformation methods themselves, the optimal choice is highly dependent on the data regime, but clear patterns emerge:

- Parametric Dominance in High-Dimensions: The parametric method REFINED is unambiguously the most robust transformation, appearing most frequently as the top performer across all tasks. Its dominance is particularly evident in high-dimensional ($d \gtrsim 1k$) regimes, e.g., MIMO-X/Y (Regression), Bioresponse (Binary), and ISOLET (Multiclass). In these complex scenarios, other parametric methods like IGTD and TINTO also perform strongly, confirming that methods capable of optimizing spatial layout are critical for high-dimensional data.
- Non-Parametric Competitiveness: In low-to-moderate dimensionality datasets, non-parametric methods are highly competitive. DM and Combination appear regularly among the best candidates (e.g., Health, Boston, CMC). Other methods like BIE and SuperTML also appear as winners in large-sample classification tasks (Nomao, Adult), demonstrating their value as computationally efficient baselines.

In summary, while hybrid models offer a consistent reduction in variance (answering **RQ3**), the choice of transformation method is a dom-



(a) Regression tasks (Normalized RMSE, lower is better).

(b) Binary classification tasks (Test Accuracy, higher is better).

(c) Multiclass classification tasks (Test Accuracy, higher is better).

**Fig. 7.** Best score per model family across (a) regression, (b) binary, and (c) multiclass datasets. The $y$-axes show normalized RMSE or test accuracy. Bars denote mean scores across seeds with error bars for variability. Text labels identify the image transformation yielding each family's best result. Dataset labels indicate samples (S) and features (F).

inant factor, with a clear trade-off: parametric methods like REFINED are the most robust default choice and excel in high-dimensional data, while efficient non-parametric methods like DM are highly competitive in simpler, low-dimensional regimes (answering **RQ4**).

### 5.4. Statistical analysis of component performance

The preceding analysis provided an empirical overview of model performance based on mean scores. To formally validate these observations, this section introduces a statistical test to determine whether the choice of vision architecture leads to significant performance differences within each transformation method.

As shown in Table 3, the Skillings-Mack test results reveal two distinct patterns. For a majority of transformation methods, including FeatureWrap, TINTO, and IGTD, the performance differences between architectures are statistically significant ($p < .05$). Within this group, a clear trend emerges where hybrid models (ViT + MLP or CNN + MLP) consistently achieve the best average rank, demonstrating a statistically validated performance advantage. Conversely, for the REFINED, Combination, and BarGraph methods, the test yields non-significant results. This indicates that for these specific transformations, there is no statistical evidence to prefer one vision architecture over another, suggesting a high degree of architectural robustness.

**Table 3**
Architecture performance by transformation method using Skillings–Mack test. Ranks are computed within each dataset (1 = best); averages reported across datasets. Lower average rank indicates better performance. Bold indicates the best architecture per method.

| Method | ViT | ViT + MLP | CNN | CNN + MLP | $T$ | $p$ | n | Sig. |
|---|---|---|---|---|---|---|---|---|
| FeatureWrap | 3.23 | **1.23** | 3.20 | 2.34 | 47.00 | 0.0000 | 22 | *** |
| TINTO | 3.17 | **1.75** | 3.00 | 2.08 | 27.47 | 0.0000 | 24 | *** |
| IGTD | 2.67 | **1.75** | 2.81 | 2.77 | 14.62 | 0.0022 | 24 | ** |
| BIE | 2.19 | **2.03** | 3.25 | 2.53 | 12.67 | 0.0054 | 18 | ** |
| DM | 3.08 | 2.50 | 2.61 | **1.81** | 12.02 | 0.0073 | 18 | ** |
| SuperTML | 2.94 | 2.24 | 2.94 | **1.88** | 11.44 | 0.0096 | 17 | ** |
| Combination | 2.81 | **2.06** | 2.28 | 2.86 | 6.78 | 0.0793 | 18 | ns |
| REFINED | 2.48 | **2.08** | 2.79 | 2.65 | 5.38 | 0.1458 | 24 | ns |
| BarGraph | **2.25** | 2.31 | 2.75 | 2.69 | 2.89 | 0.4091 | 18 | ns |

These statistical findings provide a formal basis for earlier observational claims. The consistent superiority of hybrid models in the significant group strongly corroborates the previous finding that hybrids "reliably lower variance" and offer greater stability. The test now confirms that they are often the statistically optimal choice in terms of average performance, not merely a more stable alternative. Furthermore, the architectural interchangeability for REFINED and Combination, previously identified as the most reliable parametric and practical non-parametric options, respectively, reinforces their value. Their effectiveness is not contingent on a specific architectural choice, solidifying their status as robust and flexible methods for practitioners.

Complementing the previous analysis, we now invert the perspective to evaluate which transformation method performs best for a fixed architecture. The results, presented in Table 4, show that for every architecture, the choice of transformation method yields statistically significant performance differences ($p < .01$ in all cases). A clear pattern of dominance emerges: REFINED is the statistically best-performing method for the ViT, CNN, and ViT + MLP architectures. The only exception is the CNN + MLP architecture, where the non-parametric DM achieves the top rank.

This statistical hierarchy provides robust validation for our earlier empirical findings. The consistent, top-ranked performance of REFINED across three of the four architectures solidifies its status as the most reliable default choice. The strong showing of DM, particularly its victory with the CNN + MLP model, confirms it is more than just a viable non-parametric alternative; it is the optimal choice for a specific and effective hybrid architecture. This aligns with our observation that non-parametric methods are highly competitive in certain contexts. Finally, the consistently poor ranking of FeatureWrap across all architectures provides statistical evidence for the empirical finding that it is generally less effective than other methods.

### 5.5. Aggregate performance analysis

The previous sections analyzed performance by fixing either the transformation method or the architecture. We now broaden the scope to determine if any single architecture or method demonstrates superior performance overall. To this end, we conduct two final Skillings–Mack tests, aggregating performance across all datasets.

First, we evaluate the architectures. For each dataset, every architecture is paired with its best-performing transformation method, representing a best-case scenario. The aggregated results are presented in Table 5.

The results in Table 5 provide a crucial insight that refines our earlier empirical observations. Numerically, the vision-only ViT architecture

**Table 4**
Method performance within each architecture using Skillings–Mack test. Ranks are computed within each dataset (1 = best); averages reported across datasets. Lower average rank indicates better performance. Bold indicates the best method per architecture.

| Architecture | TINTO | IGTD | REFINED | DM | BarGraph | Combination | SuperTML | FeatureWrap | BIE | $T$ | $p$ | n | Sig. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ViT | 5.50 | 6.00 | **2.67** | 4.17 | 3.42 | 3.11 | 4.71 | 6.91 | 4.92 | 73.10 | 0.0000 | 24 | *** |
| CNN | 5.31 | 5.31 | **2.94** | 4.25 | 4.33 | 3.08 | 4.29 | 6.59 | 5.53 | 49.42 | 0.0000 | 24 | *** |
| CNN + MLP | 5.02 | 5.77 | 3.50 | **3.17** | 3.97 | 4.44 | 3.74 | 6.36 | 5.44 | 46.03 | 0.0000 | 24 | *** |
| ViT + MLP | 5.21 | 4.96 | **2.94** | 4.92 | 4.36 | 4.00 | 4.71 | 5.98 | 4.89 | 26.59 | 0.0008 | 24 | *** |

**Table 5**
Aggregate performance comparison for architectures and transformation methods. Lower average ranks indicate better performance. Each architecture or method is evaluated using its optimal counterpart per dataset. Bold values denote the overall best performer within each category.

*Architecture Performance (using best transformation method)*

| Architecture | Avg Rank | Wins | Win Rate (%) |
|---|---|---|---|
| ViT | **2.29** | 9/24 | 37.5 |
| ViT + MLP | 2.35 | 6/24 | 25.0 |
| CNN + MLP | 2.67 | 5/24 | 20.8 |
| CNN | 2.69 | 5/24 | 20.8 |
| | Skillings–Mack test: $T = 2.45$, $p = 0.4844$ (n.s.) | | |

*Transformation Method Performance (using best architecture)*

| Method | Avg Rank | Wins | Win Rate (%) |
|---|---|---|---|
| REFINED | **2.79** | 12/24 | 50.0 |
| BarGraph | 3.89 | 2/24 | 8.3 |
| Combination | 4.06 | 0/24 | 0.0 |
| DM | 4.17 | 1/24 | 4.2 |
| IGTD | 4.88 | 2/24 | 8.3 |
| SuperTML | 5.15 | 2/24 | 8.3 |
| BIE | 5.22 | 2/24 | 8.3 |
| TINTO | 5.31 | 2/24 | 8.3 |
| FeatureWrap | 6.45 | 1/24 | 4.2 |
| | Skillings–Mack test: $T = 39.01$, $p < 0.001$ (***) | | |

**Table 6**
Post-hoc Nemenyi test comparing Refined against other transformation methods. Ranks are computed within each dataset (1 = best); lower ranks are better. Mean Rank Difference is defined as (*competitor* - REFINED). The null hypothesis of equal performance is rejected if the Mean Rank Difference exceeds the Critical Difference (CD) at $\alpha = 0.05$.

| Comparison Method | Mean Rank Diff. | Critical Diff. (CD) | n | Outcome ($p < .05$) |
|---|---|---|---|---|
| TINTO | 2.521 | 3.468 | 24 | Not Significant |
| IGTD | 2.083 | 3.468 | 24 | Not Significant |
| DM | 1.375 | 4.004 | 18 | Not Significant |
| BarGraph | 1.097 | 4.004 | 18 | Not Significant |
| Combination | 1.264 | 4.004 | 18 | Not Significant |
| SuperTML | 2.355 | 4.120 | 17 | Not Significant |
| FeatureWrap | 3.663 | 3.622 | 22 | **Refined is better** |
| BIE | 2.431 | 4.004 | 18 | Not Significant |

achieves the best average rank and the highest win rate, which aligns with the initial finding that vision-only models win more frequently on average across the different tasks. However, the Skillings–Mack test reveals that these differences are not statistically significant ($p = .4844$). This is a key finding: when each architecture is optimized with its best transformation method, no single architecture proves to be statistically superior on a global level. The slight edge observed for vision-only models in the empirical analysis does not translate to a statistically robust advantage. This suggests that performance is driven more by the successful pairing of a method and an architecture rather than by an inherent, universal advantage of any single architectural choice.

Having established that no single architecture holds a statistically significant advantage, we now apply the same analysis to the transformation methods to determine if a single method emerges as the overall winner.

In stark contrast to the architectural comparison, the analysis of transformation methods in Table 5 reveals a clear and statistically significant hierarchy ($p < .001$). The REFINED method is the unambiguous winner, achieving a far superior average rank of 2.79 and winning 50% of all datasets–a stark lead over all other methods. This result provides the strongest possible statistical validation for the central claim of our empirical analysis: that REFINED is the most reliable transform across regimes and should be considered the default choice. While other non-parametric methods like Combination and DM were identified as practical options, this aggregate statistical test confirms that they do not reach the same level of overall performance as REFINED when each method is optimized with its best architectural partner.

To probe the significance of this victory, we conduct a final post-hoc Nemenyi test, presented in Table 6, comparing REFINED against each competitor individually. The results add a crucial layer of nuance. While REFINED's superiority is statistically significant when compared to the weakest method, FeatureWrap, its performance advantage is not statistically significant against the other strong contenders, including IGTD, Combination, and DM. This finding does not contradict REFINED's status as the best-performing method; rather, it clarifies it. REFINED is the leader among a group of top-performing methods that are statistically indistinguishable from one another.This aligns perfectly with our empirical observations, where methods like Combination and DM were noted to be highly competitive and practical options in many regimes.

## 6. Discussion of findings and limitations

This section interprets the benchmark results from Section 5. We first provide a qualitative analysis of the synthetic image representations , connecting their visual features to the observed performance. We then synthesize the quantitative findings to answer our research questions and conclude by addressing the study's limitations and threats to validity.

### 6.1. Qualitative analysis of synthetic images

The tabular data into synthetic images paradigm is grounded in the premise that imposing a spatial organization on inherently order-free tabular data allows vision models to exploit their inductive biases. The methods benchmarked in this study pursue this objective through two complementary design philosophies, which have direct and critical implications for handling the arbitrary nature of feature order.

Parametric methods, i.e., REFINED, IGTD, and TINTO, create spatially meaningful synthetic images by learning an optimal layout in which feature relationships are encoded as spatial proximity. This process is inherently order-invariant, as it discards the original column sequence to discover a data-driven organization (see Fig. 8a–e). In contrast, non-parametric methods such as DM, BarGraph, Combination, FeatureWrap, SuperTML, and BIE produce deterministic, rule-based visualizations whose spatial structure depends directly on the initial feature order (see Fig. 8f–h). Although they lack intrinsic spatial semantics, these methods still generate consistent visual patterns that vision models can exploit.

Beyond feature order, the composition of the tabular data, particularly the balance between numerical and categorical attributes, strongly

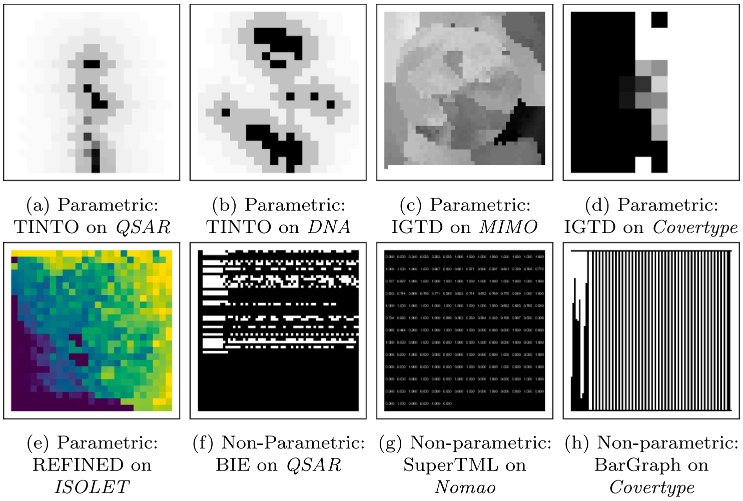

(a) Parametric: TINTO on *QSAR*
(b) Parametric: TINTO on *DNA*
(c) Parametric: IGTD on *MIMO*
(d) Parametric: IGTD on *Covertype*
(e) Parametric: REFINED on *ISOLET*
(f) Non-Parametric: BIE on *QSAR*
(g) Non-parametric: SuperTML on *Nomao*
(h) Non-parametric: BarGraph on *Covertype*

**Fig. 8.** Visual comparison of tabular data into synthetic image transformations, both parametric and non-parametric methods across representative datasets.

shapes the visual properties of the resulting synthetic images. For high-dimensional, fully numerical datasets, e.g., MIMO and ISOLET (see Fig. 8c and e), parametric transformations yield information-dense and smoothly structured representations. In datasets dominated by categorical variables, such as Covertype, one-hot encoding expands each category into high-dimensional binary vectors, introducing sparsity that disrupts spatial coherence. This results in discrete, textureless patterns across both parametric and non-parametric methods (see Fig. 8h and d), with non-parametric approaches yielding rigid, binary-like layouts and parametric methods showing reduced smoothness. However, the preceding analysis shows that a high proportion of categorical features does not systematically favor either classical or neural pipelines. Nevertheless, the exploration of alternative embedding techniques could mitigate this limitation by replacing sparse one-hot vectors with dense, continuous representations that preserve category semantics. Incorporating these embeddings before the transformation would yield smoother spatial organization and more informative visual patterns for vision models.

To address **RQ2**, we compare the computational efficiency and scalability of parametric and non-parametric encoding families (see Section 3.3). Parametric methods exhibit a clear dimensionality-reduction effect that enables scalable representation of high-dimensional data. Unlike non-parametric mappings, whose synthetic image size grows with the number of input features, parametric methods such as TINTO maintain a fixed resolution, while others, like REFINED and IGTD, produce synthetic images whose dimensions scale efficiently with the square root of the number of features. This behavior becomes especially evident in high-dimensional datasets: the MIMO dataset (see Fig. 8c) with 1600 numerical features is represented as a compact 40 × 40 image. Similarly, the DNA dataset (Fig. 8b) with 180 categorical features is projected onto a 20 × 20 grid, avoiding the dimensionality explosion that one-hot encoding would otherwise produce.

This compression, however, incurs high computational cost during the fitting phase, especially for REFINED (∼20,000 s in high-dimensional datasets such as Bioresponse), followed by IGTD (∼700 s) and TINTO ( < 60 s). Non-parametric methods, with negligible fitting time, are faster on large datasets but less efficient as dimensionality increases due to image size growth. While the transformation phase scales with sample size for all methods, it could be parallelized, mitigating runtime on large datasets like Covertype. In summary, parametric methods offer compact and scalable representations at higher computational cost, with TINTO achieving the best trade-off between efficiency and scalability. Non-parametric methods are computationally light but unsuitable for high-dimensional data. These results define the central conclusion of **RQ2**.

### 6.2. Synthesis of findings

Our comprehensive experimental evaluation, spanning empirical analysis across regression, binary, and multiclass tasks, and culminating in a series of formal statistical tests, reveals a clear and multi-faceted view of what drives performance in synthetic image-based tabular learning. The findings tell a cohesive story, where initial observational trends are consistently validated, refined, and deepened by statistical scrutiny.

The initial exploration established that performance is governed not by task type, but by the data regime, defined by sample size ($N$) and feature dimensionality ($d$). This empirical performance map suggested that classical models excel in data-rich, low-dimensional settings, while neural pipelines hold an advantage when data is scarce or dimensionality is extreme. It also identified two key patterns: the consistent variance-reduction effect of hybrid models, which reliably narrowed error bars even when not winning on mean performance, and the emergence of REFINED as the most reliable transformation method across diverse scenarios.

The subsequent statistical analysis provided a rigorous validation of these observations, leading to a central, striking asymmetry: the choice of transformation method is of critical and statistically significant importance, whereas the choice of vision architecture is a secondary, context-dependent decision. Our aggregate tests show that no single architecture, i.e., ViT, CNN, or their hybrid variants, is superior overall; the empirical observation of a slight edge for vision-only models does not hold up to formal scrutiny. Instead, the optimal architecture is contingent on the transformation method. For many methods, hybrid models proved to be the statistically superior choice, providing a formal basis for the empirical finding that they consistently improve model stability. For top-tier, robust methods like REFINED, however, the architectural choice was not statistically significant, affording practitioners valuable flexibility.

Ultimately, performance is overwhelmingly dictated by the transformation. Here, REFINED was statistically confirmed as the best-performing method, achieving the highest win rate and the best average rank. However, a final post-hoc analysis added a crucial nuance: while REFINED is the clear leader, its performance is not statistically superior to most of the other methods. As shown in Table 6, its advantage over strong contenders like IGTD, Combination, DM, and BarGraph is not statistically significant. This result confirms our empirical observation that other methods like IGTD and Combination are strong competitors. It demonstrates that while REFINED has the best average performance, it belongs to a group of top-tier methods that are statistically equivalent.

The key takeaway for practitioners is therefore a clear hierarchy of decisions. The primary and most critical choice is the transformation method; this involves selecting from the top-performing methods–with REFINED as the safest and most consistently high-performing default. The secondary choice of architecture, while less critical for achieving peak performance, offers a strategic opportunity to enhance model stability by opting for a hybrid variant.

### 6.3. *Threats to validity*

Our study was designed with specific methodological choices that define the scope of our findings and provide clear directions for future research. Our focus is on the tabular data into synthetic image conversion paradigm, meaning our conclusions on neural performance do not necessarily extend to other deep tabular architectures, such as those based on contextual embeddings. For preprocessing, we standardized on one-hot encoding as a common baseline; while effective, we acknowledge that alternative strategies like entity embeddings could yield different outcomes, particularly for high-cardinality features.

Similarly, our benchmark prioritizes a rigorous, unified optimization of the downstream model architectures. Consequently, the upstream transformation methods were applied using their standard, recommended hyperparameters. While an exhaustive search over transformation-specific parameters could yield further marginal gains, our approach isolates the impact of the transformation methodology itself. Furthermore, non-parametric methods whose synthetic image dimensions scale with the number of features (i.e., BarGraph, DM, Combination, SuperTML and BIE) were excluded from high-dimensional datasets where their native image size becomes computationally expensive. While these large images could have been downsampled, we opted to evaluate them at their native resolution to avoid introducing a potentially confounding information-loss step, thereby limiting their application to low- and mid-dimensional data in this study.

Methodologically, we also chose not to employ explicit class-balancing techniques in order to evaluate each model's baseline performance on the natural data distribution. We recognize this may favor models inherently more robust to skewed distributions, e.g., GBDT, and our results should be interpreted in this context. Finally, as with any benchmark, our conclusions are drawn from a finite set of 24 datasets, and the performance patterns identified may shift when applied to domains with different underlying data-generating processes.

## 7. Conclusion and open challenges

This work presents a systematic benchmark of tabular data into synthetic image methods and neural architectures for tabular data. By evaluating nine transformation methods across 24 datasets under a unified and rigorous hyperparameter optimization framework, we provide a comprehensive view of the factors that drive performance.

Our findings show that performance is determined primarily by the data regime. Neural pipelines excel in data-scarce or high-dimensional settings, whereas classical models dominate in data-rich, low-dimensional ones. A clear asymmetry emerges: the choice of transformation method exerts a statistically and practically greater influence on performance than the choice of vision architecture, which remains context-dependent. Although aggregate differences between architectures such as ViT and CNN are not statistically significant, hybrid designs (ViT + MLP, CNN + MLP) excel in small datasets and consistently reduce predictive variance across all regimes, confirming the advantage of combining vision-based models with MLPs.

Among transformation methods, REFINED demonstrates the highest robustness across tasks and architectures, leading a top tier of statistically comparable approaches that includes IGTD, Combination, and DM. Parametric methods like REFINED offer compact and spatially meaningful representations but are computationally demanding due to their optimization-based fitting phase. In contrast, non-parametric methods are lightweight yet scale poorly with dimensionality, as image size grows linearly with the number of features.

Future work should focus on optimizing scalability and interpretability. Integrating categorical features, via embeddings or distance-based metrics, is a key step toward broader applicability. Enhancing the interpretability of dense transformations like REFINED and improving parametric pipelines for large-scale data are also priorities. Finally, our hybrid models use basic feature concatenation; exploring more advanced fusion strategies (e.g., cross-attention or Transformer-encoder) inspired by multimodal learning could further unlock the potential of synthetic image-based models for tabular data [19]. Additionally, integrated transformation strategies, where synthetic image generation is learned jointly with the model, offer a promising direction for adaptive spatial encoding.

## CRediT authorship contribution statement

**Jiayun Liu:** Writing – review & editing, Writing – original draft, Visualization, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization; **Manuel Castillo-Cara:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Methodology, Investigation, Formal analysis, Conceptualization; **Raúl García-Castro:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization.

## Data availability

All the resources used in this scientific experiment are recorded and can be found at Zenodo [37]. In addition, a dedicated benchmark webpage is maintained[3] [38], where results and comparative analyses across a growing set of datasets, transformation methods, and neural architectures are continuously updated. This page aims to foster reproducibility and transparency while providing researchers with a reference point for further exploration and evaluation in this area.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgments

The research leading to these results has been funded by the European Commission H2020 project “Construction-phase diGItal Twin mOdel (COGITO)” under contract #958310, by grant PID2023-150515OB-I00 from the Spanish Government, partially covered with funds from the Recovery and Resilience Facility (RRF), and by the Madrid Government (Comunidad de Madrid-Spain) under the Multiannual Agreement with the Universidad Politécnica de Madrid in the Excellence Programme for University Teaching Staff, in the context of the V PRICIT (Regional Programme of Research and Technological Innovation).

The authors gratefully acknowledge the Universidad Politécnica de Madrid (www.upm.es) for providing computing resources on the IPTC-AI innovation Space AI Supercomputing Cluster. A language-model-assisted editing tool was used exclusively to improve the clarity and readability of the manuscript. All methodological, analytical, and scientific contributions were entirely conceived, implemented, and validated by the authors. Open access funding for this publication was provided by Universidad Nacional de Educación a Distancia (UNED).

[3] https://oeg-upm.github.io/TINTOlib/

## References

[1] W. Bousselham, A. Boggust, S. Chaybouti, H. Strobelt, H. Kuehne, LeGrad: an explainability method for vision transformers via feature formation sensitivity, arXiv:2404.03214. (2024).

[2] M. Dreyer, E. Purelku, J. Vielhaben, W. Samek, S. Lapuschkin, PURE: turning polysemantic neurons into pure features by identifying relevant circuits, arXiv:2404.06453. (2024).

[3] F.J. Lara-Abelenda, D. Chushig-Muzo, P. Peiro-Corbacho, V. Gómez-Martínez, A.M. Wägner, C. Granja, C. Soguero-Ruiz, Transfer learning for a tabular-to-image approach: a case study for cardiovascular disease prediction, J. Biomed. Inform. 165 (2025) 104821. https://doi.org/10.1016/j.jbi.2025.104821

[4] M.I. Iqbal, M.S.H. Mukta, A.R. Hasan, S. Islam, A dynamic weighted tabular method for convolutional neural networks, IEEE Access 10 (2022) 134183–134198. https://doi.org/10.1109/ACCESS.2022.3231102

[5] B. Van Breugel, M. Van Der Schaar, Position: why tabular foundation models should be a research priority, in: R. Salakhutdinov, Z. Kolter, K. Heller, A. Weller, N. Oliver, J. Scarlett, F. Berkenkamp (Eds.), Proceedings of the 41st International Conference on Machine Learning, 235 of *Proceedings of Machine Learning Research*, PMLR, 2024, pp. 48976–48993. https://proceedings.mlr.press/v235/van-breugel24a.html.

[6] M. Neto, L. Neto, R. da Silva, S. Endo, P. Takako, A comparative analysis of converters of tabular data into image for the classification of arboviruses using convolutional neural networks, PLoS ONE 18 (12) (2023) 1–23. https://doi.org/10.1371/journal.pone.0295598

[7] V. Borisov, T. Leemann, K. Seßler, J. Haug, M. Pawelczyk, G. Kasneci, Deep neural networks and tabular data: a survey, IEEE Trans. Neural Netw. Learn. Syst. 35 (6) (2024) 7499–7519. https://doi.org/10.1109/TNNLS.2022.3229161

[8] R. Shwartz-Ziv, A. Armon, Tabular data: deep learning is not all you need, Inform. Fusion 81 (2022) 84–90.

[9] A. Damri, M. Last, N. Cohen, Towards efficient image-based representation of tabular data, Neural Comput. Appl. 36 (2) (2024) 1023–1043. https://doi.org/10.1007/s00521-023-09074-y

[10] O. Bazgir, S. Ghosh, R. Pal, Investigation of REFINED CNN ensemble learning for anti-cancer drug sensitivity prediction, Bioinformatics 37 (2021) 42–50.

[11] G.W. Kyro, R.I. Brent, V.S. Batista, HAC-Net: a hybrid attention-based convolutional neural network for highly accurate protein-ligand binding affinity prediction, J. Chem. Inf. Model. 63 (7) (2023) 1947–1960. PMID: 36988912, https://doi.org/10.1021/acs.jcim.3c00251

[12] R. Yan, M.T. Islam, L. Xing, Interpretable discovery of patterns in tabular data via spatially semantic topographic maps, Nat. Biomed. Eng. 9 (2024) 1–12.

[13] A. Sharma, E. Vans, D. Shigemizu, K.A. Boroevich, T. Tsunoda, DeepInsight: a methodology to transform a non-image data to an image for convolution neural network architecture, Sci. Rep. 9 (1) (2019) 11399. https://doi.org/10.1038/s41598-019-47765-6

[14] O. Bazgir, R. Zhang, S.R. Dhruba, R. Rahman, S. Ghosh, R. Pal, Representation of features as images with neighborhood dependencies for compatibility with convolutional neural networks, Nat. Commun. 11 (2020) 4391.

[15] Y. Zhu, T. Brettin, F. Xia, A. Partin, M. Shukla, H. Yoo, Y.A. Evrard, J.H. Doroshow, R.L. Stevens, Converting tabular data into images for deep learning with convolutional neural networks, Sci. Rep. 11 (2021) 11325.

[16] H. Tang, X. Yu, R. Liu, T. Zeng, Vec2Image: an explainable artificial intelligence model for the feature representation and classification of high-dimensional biological data by vector-to-image conversion, Brief. Bioinform. 23 (2) (2022) bbab584. https://doi.org/10.1093/bib/bbab584

[17] R. Talla-Chumpitaz, M. Castillo-Cara, L. Orozco-Barbosa, R. García-Castro, A novel deep learning approach using blurring image techniques for bluetooth-based indoor localisation, Inform. Fusion 91 (2023) 173–186.

[18] V. Gómez-Martínez, F.J. Lara-Abelenda, P. Peiro-Corbacho, D. Chushig-Muzo, C. Granja, C. Soguero-Ruiz, LM-IGTD: a 2D image generator for low-dimensional and mixed-type tabular data to leverage the potential of convolutional neural networks, arXiv:2406.14566. (2024).

[19] M. Castillo-Cara, J. Martínez-Gómez, J. Ballesteros-Jerez, I. García-Varea, R. García-Castro, L. Orozco-Barbosa, MIMO-based indoor localisation with hybrid neural networks: leveraging synthetic images from tidy data for enhanced deep learning, IEEE J. Sel. Top. Signal Process. (2025) 1–13. https://doi.org/10.1109/JSTSP.2025.3555067

[20] B. Bischl, G. Casalicchio, M. Feurer, P. Gijsbers, F. Hutter, M. Lang, R. Gomes Mantovani, J. van Rijn, J. Vanschoren, OpenML benchmarking suites, in: J. Vanschoren, S. Yeung (Eds.), Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks, 1, 2021.

[21] S.F. Fischer, L.H.M. Feurer, B. Bischl, OpenML-CTR23 – a curated tabular regression benchmarking suite, in: AutoML Conference 2023 (Workshop), 2023.

[22] J. Zhang, J. Tian, M. Li, J.I. Leon, L.G. Franquelo, H. Luo, S. Yin, A parallel hybrid neural network with integration of spatial and temporal features for remaining useful life prediction in prognostics, IEEE Trans. Instrum. Meas. 72 (2022) 1–12.

[23] S.M. Zandavi, D. Liu, V. Chung, A. Anaissi, F. Vafaee, Fotomics: fourier transform-based omics imagification for deep learning-based cell-identity mapping using single-cell omics profiles, Artif. Intell. Rev. 56 (7) (2023) 7263–7278. https://doi.org/10.1007/s10462-022-10357-4

[24] T.N. Wolf, S. Pölsterl, C. Wachinger, DAFT: a universal module to interweave tabular data and 3D images in CNNs, Neuroimage 260 (2022) 119505. https://doi.org/10.1016/j.neuroimage.2022.119505

[25] J.-P. Jiang, S.-Y. Liu, H.-R. Cai, Q. Zhou, H.-J. Ye, Representation learning for tabular data: a comprehensive survey, arXiv:2504.16109. (2025).

[26] E. Lee, M. Nam, H. Lee, Tab2vox: CNN-based multivariate multilevel demand forecasting framework by tabular-to-voxel image conversion, Sustainability 14 (2022) 11745.

[27] H. Liu, K. Simonyan, Y. Yang, DARTS: differentiable architecture search, in: International Conference on Learning Representations, 2019.

[28] A. Sharma, D. Kumar, Classification with 2-D convolutional neural networks for breast cancer diagnosis, Sci. Rep. 12 (2022) 21857.

[29] B. Sun, L. Yang, W. Zhang, M. Lin, P. Dong, C. Young, J. Dong, SuperTML: two-dimensional word embedding for the precognition on structured tabular data, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE Computer Society, 2019, pp. 2973–2981.

[30] Z. Li, Z. Qin, K. Huang, X. Yang, S. Ye, Intrusion detection using convolutional neural networks for representation learning, in: D. Liu, S. Xie, Y. Li, D. Zhao, E.-S.M. El-Alfy (Eds.), Neural Information Processing, Springer International Publishing, Cham, 2017, pp. 858–866.

[31] N. Briner, D. Cullen, J. Halladay, D. Miller, R. Primeau, A. Avila, R. Basnet, T. Doleck, Tabular-to-image transformations for the classification of anonymous network traffic using deep residual networks, IEEE Access 11 (2023) 113100–113113. https://doi.org/10.1109/ACCESS.2023.3323927

[32] J. Halladay, D. Cullen, N. Briner, D. Miller, R. Primeau, A. Avila, W. Watson, R. Basnet, T. Doleck, BIE: binary image encoding for the classification of tabular data, J. Data Sci. 23 (1) (2025) 109–129. https://doi.org/10.6339/24-JDS1122

[33] A. Sharma, Y. Lopez, S. Jia, A. Lysenko, K. Boroevich, T. Tsunoda, Enhanced analysis of tabular data through Multi-representation DeepInsight, Sci Rep 14 (2024) 12851. https://doi.org/10.1038/s41598-024-63630-7

[34] M. Castillo-Cara, R. Talla-Chumpitaz, R. García-Castro, L. Orozco-Barbosa, TINTO: converting tidy data into image for classification with 2-dimensional convolutional neural networks, SoftwareX 22 (2023) 101391.

[35] J. Liu, D. González-Fernández, M. Castillo-Cara, R. García-Castro, TINTOlib: A Python library for transforming tabular data into synthetic images for deep neural networks, SoftwareX, 32, 102444, 2025, https://doi.org/10.1016/j.softx.2025.102444

[36] M. Castillo-Cara, J. Liu, TINTOlib documentation, 2024, (Link: https://tintolib.readthedocs.io/en/latest/). [Online: accessed 18-oct-2025].

[37] J. Liu, M. Castillo-Cara, R. García Castro, TINTOlib benchmark v0.0.2, 2025, https://doi.org/10.5281/zenodo.15607155

[38] J. Liu, D. González-Fernández, M. Castillo-Cara, R. García-Castro, TINTOlib: A Python library for transforming tabular data into synthetic images for deep neural networks, SoftwareX, 32, 102444, 2025, https://doi.org/10.1016/j.softx.2025.102444