DETRmining the Cosmos: A Transformer-Based Approach to Galaxy Morphology Detection

Kumar Chandra
Stanford University
kumarc@stanford.edu —

Max Rodriguez
Stanford University
maxrod@stanford.edu —

Renn Su
Stanford University
rrsu@stanford.edu

Abstract

We present an end-to-end method for classifying galaxy morphology in wide-field astronomical surveys. Our approach uses a multistep detection transformer (DETR) architecture and integrates a convolutional or self-supervised DINO-based backbone to localize and classify Ritchey-Chrétien telescopic galaxy imagery in a single pass. We apply this pipeline to three benchmark datasets: DemoRings (binary ring vs. non-ring), GalaxyMNIST (four-class morphology), and Galaxy Zoo 2 (seven-class morphology) — leveraging Hungarian bipartite matching for joint box and class supervision. We show that self-supervised DINO features accelerate convergence and improve classification, while DETR's end-to-end formulation obviates heuristic region proposals.

1. Introduction

Galaxy morphology is a critical measure in observational astronomy and astrophysics, providing insight into the physics behind galaxy formations and related galactic constituents [8]. Although there is an abundance of publicly available data from telescopes (e.g. Hubble, James Webb), analysis and interpretation of the galaxy imaging rely on slow and labor-intensive processes rooted in domain knowledge and attention to detail [6, 8]. Interpreting the density of data is a bottleneck of astronomical research, motivating the search for automated methods to detect and label galaxy morphology. Computer vision and deep vision methods have shown promising results in the detection of celestial objects within interpreting telescopic and multi-wavelength imaging, providing the possibility for accurate and efficient galaxy classification.

Although recent advances in machine learning have led to significant progress in galaxy detection, current methods still face limitations in accurately classifying galaxies by type [8, 2, 9]. In particular, many existing approaches rely on traditional convolutional pipelines and do not incorporate attention-based mechanisms into their classification frameworks. With the rise of deep learning, especially transformer-based architectures, there is strong potential to enhance galaxy classification by leveraging models that can more effectively capture long-range spatial dependencies.

Our model takes sets of images, each containing one primary galaxy in need of classification. Trained using ground truth galaxy description labels and ground truth boxes containing the galaxy through primary component/object segmentation, our DETR pipeline returns predicted labels and precise locations (in the form of bounding boxes) for unlabeled novel holdout image data galaxies.

The input to our algorithm is a single-frame, RGB-encoded galaxy image (containing exactly one primary galaxy). We then use a DETR-based model (a CNN or self-supervised DINO backbone followed by a transformer decoder) to output a predicted morphological class (e.g., ring vs. non-ring, smooth vs. spiral subtype, etc.) and a normalized bounding box around the primary galaxy.

Through extensive hyperparameter optimization (tuning learning rates, batch sizes, query counts, and scheduler strategies) and ablation experiments, we show that DINO-derived features accelerate convergence and boost classification accuracy, while DETR's end-to-end detection eliminates the need for heuristic proposal steps. In our final experiments, we achieve over 97% accuracy on hold-out splits of GalaxyMNIST and outperform a ResNet-18 baseline by 25.3% in overall accuracy, demonstrating the effectiveness of self-supervised features in astronomical object detection.

2. Related Works

2.1. GalaxyZoo and Large-Scale Survey Pipelines of Astronomical Objects

The Galaxy Zoo initiative began in 2007 as a crowd-sourced effort to classify galaxy morphologies by enlisting volunteer citizen scientists [1]. In Galaxy Zoo 2 (GZ2), over 300,000 Sloan Digital Sky Survey (SDSS) images were presented to volunteers, who collectively provided detailed morphological vote fractions across more than a dozen questions per galaxy (e.g., "Is it smooth or featured?"; "How many spiral arms?") [13]. Each galaxy's final label is determined by aggregating these votes, yielding robust but inherently slow, manual annotations.

While volunteer labeling produces high-quality morphological catalogs (e.g., over 100 million votes in GZ2), the manual nature of these campaigns limits turnaround time and scalability as survey data volumes grow [7]. For example, labeling a new data release can take months to years of continuous volunteer effort, delaying scientific analysis and follow-up studies. Our work addresses this limitation by providing an end-to-end, transformer-based pipeline that automates both localization and classification of galaxies, thereby reducing reliance on manual vote fractions and enabling near-real-time morphological catalogs for future wide-field surveys.

2.2. Machine Learning and CNNs in Galaxy Morphology Detection

Automating galaxy morphology classification has long been a critical challenge in observational astronomy. Early efforts combined handcrafted feature-extraction with classical machine-learning models (e.g., random forests, SVMs), but these methods struggled when faced with large, noisy datasets [1]. Previous works have used traditional machine learning techniques and convolutional neural networks (CNNs) in galaxy morphology ([2], [6]). Barchi et al. (2020) compare traditional ML against deep-learning approaches on galaxy morphology and find that once a few thousand labeled examples are available, CNNs consistently achieve higher accuracy than feature-based pipelines. Furthermore, aggressive data augmentation (e.g. rotations, flips, brightness/contrast jitter) significantly improves CNN robustness on smaller galaxy datasets [6]. However, previous CNN approaches face limitations in being computationally expensive at a survey scale [11]. In the domain-based context of galaxy detection, these approaches face difficulties in identifying or classifying distant or occluded features.

2.3. Developments in Transformer-Based Object Detection

Object detection with transformers (DETR) has shown promise in a variety of applications and classification tasks. Carion et al. (2020) introduce an end-to-end object-detection framework that replaces region-proposal and non-max-suppression steps with a single transformer encoder–decoder [3]. Jia et al. (2022) extend transformer architectures (similar to DETR) to the specialized task of finding gravitationally lensed arcs in galaxy-cluster imaging [9]. The results suggest that transformer-based object detectors are promising for applications in astronomy, as they outperform Hough transforms and CNN-based architectures in several cluster fields. Building on this momentum, our work represents the first demonstration—across multiple galaxy morphology benchmarks—of integrating a self-supervised DINO backbone into a DETR framework for telescopic imagery. DINO ("Distillation with No Labels") is a self-supervised Vision Transformer (ViT) pretraining method that employs a student–teacher distillation mechanism without requiring any annotated data [4].

3. Methods

We propose a method for multi-class galaxy classification using the DEtection TRansformer (DETR) framework. Our pipeline begins by passing input images through a self-supervised DINO or a convolutional neural network (CNN) backbone to extract dense feature tensors, where each patch (or pixel) is represented by a high-dimensional tensor of self-supervised or convolutional channel responses. These feature tensors are then passed into a multi-head self-attention transformer module, which uses a fixed set of learned object queries to predict class labels and bounding boxes in parallel. To supervise these predictions, we apply the Hungarian Matching algorithm to associate each predicted query with a corresponding ground truth object. Ground truth bounding boxes are derived from segmentation masks using a connected-component-based box extractor, allowing us to provide supervision for both localization and classification. This end-to-end pipeline enables joint learning of spatial and semantic representations for robust galaxy detection and classification.

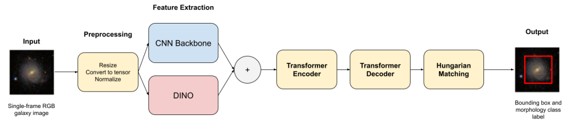

Figure 1. End-to-end DETR-based galaxy classification pipeline.

3.1. Transformers for Galaxy Queries

We adopt a transformer-based architecture inspired by DEtection TRansformers (DETR). First, we flatten the feature map $x \in \mathbb{R}^{B\times C\times H'\times W'}$ produced by either the CNN backbone (where C is the number of convolutional output channels) or the DINO backbone (where C is the projected embedding dimension, e.g., 256). Here, B is the batch size, and H', W' are the spatial dimensions of the feature map. We reshape x into a sequence of tokens $X \in \mathbb{R}^{B\times S\times D}$, where $S = H' \times W', \qquad D = C$.

Positional encodings—constructed from sine-cosine functions—are added to X so that each token retains explicit spatial context when passed through the transformer layers.

The transformer decoder uses a fixed set of N learnable object queries $Q \in \mathbb{R}^{N\times D}$, motivated by the goal that each query will attend to a distinct object or region in the original image. In practice, N can be larger than the maximum number of objects present; unmatched queries are treated as "no object."

Our architecture consists of:

- A three-layer transformer encoder, each layer containing multi-head self-attention and a feedforward network. The encoder processes the entire token sequence X to produce globally contextualized features.
- A three-layer transformer decoder, where each layer first applies self-attention over the N object queries, then cross-attention from those queries to the encoder outputs. This cross-attention step allows each query to aggregate information from all S token positions.

After decoding, we obtain a set of object-specific representations $Z \in \mathbb{R}^{B \times N \times D}$. Each slice $Z_{b,i} \in \mathbb{R}^{D}$ (for batch index b and query index i) encodes both spatial and semantic information for a candidate object. We then feed Z into parallel prediction heads:

- A class head that produces logits over $K + 1$ classes (including a "no-object" category).
- A bounding-box head that regresses a normalized $(cx,\, cy,\, w,\, h)$ box for each query.

Training is supervised end-to-end via Hungarian bipartite matching between the N predictions and the ground truth objects.

3.1.1 CNN Feature Extraction

We use a convolutional neural network (CNN) backbone to extract semantically rich, spatially compressed representations of telescopic images. The CNN accepts inputs $x \in \mathbb{R}^{B \times 3 \times H \times W}$ and produces feature maps $x' \in \mathbb{R}^{B \times C \times \frac{H}{4} \times \frac{W}{4}}$, where B is the batch size and $C = 256$ is the output channel dimension. In other words, the spatial resolution is reduced by a factor of four in each dimension.

The CNN architecture is as follows:

1. **Conv1:** $\mathrm{Conv2d}(3 \rightarrow 64, \mathrm{kernel} = 7, \mathrm{stride} = 2, \mathrm{padding} = 3)$, followed by BatchNorm and ReLU.
2. **MaxPool:** $\mathrm{MaxPool2d}(\mathrm{kernel} = 2, \mathrm{stride} = 2)$.
3. **Conv2:** $\mathrm{Conv2d}(64 \rightarrow 128, \mathrm{kernel} = 3, \mathrm{stride} = 1, \mathrm{padding} = 1)$, followed by BatchNorm and ReLU.
4. **Conv3:** $\mathrm{Conv2d}(128 \rightarrow 256, \mathrm{kernel} = 3, \mathrm{stride} = 1, \mathrm{padding} = 1)$, followed by BatchNorm and ReLU.

Because the network is shallow, we used ReLU for all activations and did not observe any vanishing-gradient issues, so we did not experiment with alternatives like leaky ReLU. We chose a larger kernel (7) and stride (2) for the first convolution to quickly reduce spatial dimensions without losing important features. For Conv2 and Conv3, we used kernel size 3, stride 1, and padding 1 to preserve the $\frac{H}{4} \times \frac{W}{4}$ resolution. The single MaxPool layer after Conv1 further halves the feature map to $\frac{H}{4} \times \frac{W}{4}$.

These feature maps preserve object-level structures in a compact format and match the high-dimensional input requirements of DETR. We selected this backbone architecture to balance data compactness with the resolution needed for precise localization in our DETR pipeline.

3.1.2 DINO Backbone Architecture

We use DINO as an alternate backbone architecture in our DETR pipeline due to its anchor-guided query initialization and denoising-based training. Previous literature has indicated that DINO allow for more localized initialization and more accurate training in earlier epochs. For noisy datasets such as our SDSS-derived galaxy imagery, DINO reduces the number of epochs needed to converge even as astronomical imaging quality is affected by blurs and occlusions.

Our DINO Backbone is a Vision Transformer (ViT) self-supervised on the ImageNet dataset (1.2M images) without using its ground truth labels. DINO uses contrastive learning methods such that the loss is acquired using a momentum-updated "student-teacher" mechanism as opposed to using traditional class labels. The DINO pretraining allows ViT patch embeddings a recognition of fine-grained textures, edges, and high-level semantics before downstream fine-tuning. After our image preprocessing, we take an input tensor of $x \in \mathbb{R}^{3 \times 224 \times 224}$, indicating 3 RGB channels and an image size of 224×224. (Note: any 64×64 images from DemoRings and MNIST are upscaled to 224.) The DINO Backbone outputs a feature tensor of $F_{\mathrm{DINO}} \in \mathbb{R}^{512 \times H' \times W'}$, representing the embedding dimension and a downsampled spatial grid prepared for the rest of our transformer pipeline. Due to limitations in computing power, we base our weights on the self-supervised model by Caron et al. 2021 [4].

3.2. Proposed Baseline: ResNet-18 Galaxy Morphology Classification

For our baseline metric, we use an architecture based on ResNet-18. ResNet-18 is a well-documented and lightweight model featuring ImageNet-pretrained weights. As a computationally feasible and 18-layer architecture, ResNet-18 is widely used in visual classification and detection tasks. In our specific use-case, ResNet-18 serves as a reliable option to use to represent traditional CNNs in contrast to our transformer-based pipelines. Previous literature in galaxy morphology have used ResNet-18 and other ResNet variants as a baseline metric. [12, 5].

The ResNet-18 classifies maps each image input of our galaxy datasets to a predicted class through global average pooling and a linear layer. We take an input layer of $x \in \mathbb{R}^{3\times H\times W}$ and downsize to a feature map of size $x \in \mathbb{R}^{2\times 2\times 512}$. A final average-pool reduces this to a 512-dim vector, which is mapped to the K morphology classes.

3.3. Hungarian Matching Algorithm

The classification and box-regression heads of our DETR transformer output prediction tensors of size $[B, \text{num\_queries}, \text{num\_classes}, \text{box\_dims}]$, where num_classes varies by dataset and box_dims $= 4$ (corresponding to the object's center coordinate (c_x, c_y) and width/height (w, h), as described in Section **??**). Queries are assigned to ground-truth boxes in a way that minimizes the total SetCriterion loss. Since there is only one ground-truth box per image in the Demo Rings dataset, the matching problem reduces to identifying the single transformer query that best matches the lone labeled object in each image.

Our loss function and matching procedure follow Carion *et al.* [3].

For each possible query-to-ground-truth assignment, DETR computes a combined cost consisting of three components: classification cost, L1 distance cost, and GIoU cost. The classification cost is computed by applying softmax to the predicted class logits for each query and then taking the negative log-probability assigned to the true class label. Because our dataset provides exactly one ground-truth label ("ring galaxy"), we compute this cost only for that class (and not over all classes).

Next, to evaluate spatial alignment, DETR converts predicted and ground-truth boxes from center format (c_x, c_y, w, h) to corner coordinates $(x_{\min}, y_{\min}, x_{\max}, y_{\max})$. We then compute the L1 distance between each pair of matched boxes (measuring absolute difference in center and scale) and calculate the negative GIoU as an additional cost that rewards precise overlap. The Hungarian algorithm finds the assignment s that minimizes the total sum of these costs over all N queries.

Implementation details for the matching and loss computation are provided in Section 3.4.

3.4. Set Criterion

For our SetCriterion function, or DETR loss function, we calculate loss after the Hungarian matching function has assigned queries to ground truth objects.

Cross-entropy loss is first computed for the ground-truth class scores using the softmax probabilities from the queries: -log(p[ground-truth class]). Next, L1 loss is used to measure the mean absolute error between the predicted query box coordinates and the corresponding ground truth box.

Finally, the total SetCriterion loss is obtained by summing the classification and bounding box losses across all examples, weighting them by class_coef = 1 and bbox_coef = 5, and averaging the result over the batch size. Semantically, a higher weight is given to the bounding box loss coefficient in order to prioritize locating objects in space over correct classification labels.

4. Dataset and Features

The telescopic imagery from our datasets was taken as part of the Sloan Digital Sky Survey (SDSS). Images were annotated and categorized through Galaxy Zoo citizen science campaigns. For our paper, we specifically use the DemoRings, GalaxyMNIST [14], and GalaxyZoo2 datasets [12].

Dataset Name	Train Size	Test Size	Resolution (px)	Source
DemoRings	800	200	64 × 64	Walmsley et al. (2024)
GalaxyMNIST	8 000	2 0	64 × 64	Willett et al. (2022)
Galaxy Zoo 2	180 000	30 000	424 × 424	Sichkar (2020)

Table 1. Summary of the three galaxy imagery datasets used in our experiments.

For initial development and evaluation of our DETR pipeline, we use the Demo Rings and MNIST test subsets of the GZ2 dataset, which includes binary and four-class classification tasks respectively. We use the Galaxy Zoo 2 (GZ2) dataset for training our final model. GZ2 contains over 300,000 galaxy images annotated through crowd-sourced visual classification [13]. More specifically, each galaxy image in the GZ2 dataset was given a description label based on an at least 17 person majority vote. MNIST and demo rings has already been pre-processed to match the majority vote label with the respective image as ground truth, while the raw gz2 dataset still contains the vote counts for each possible label. Each image features a single primary galaxy, and binary labels were assigned based on a majority vote from Galaxy Zoo citizen scientist participants. Our goal is to accurately classify the morphology of each galaxies while leveraging object detection to localize the primary object in each image.

4.1. Data Preprocessing and Augmentation

4.1.1 Pre-processing

We first resize every image to a fixed spatial dimension—namely 64 × 64 pixels—so that the batch of images can be stacked into a single tensor without shape mismatches. Uniform resizing also ensures that downstream convolutional layers see the same input resolution on every forward pass. After resizing, we call ToTensor(), which converts an image or NumPy array in the range [0, 255] into

Dataset Name	# Classes	Class Details
DemoRings	2	{no-ring, ring}
GalaxyMNIST	4	{smooth & round, smooth & cigar-shaped, edge-on disk, unbarred spiral}
Galaxy Zoo 2	7	{smooth & round, smooth & cigar-shaped, edge-on disk, unbarred spiral, barred spiral, smooth inbetweeen, featured without bar or spiral}

Table 2. Overview of each galaxy dataset: number of discrete classes and detailed label information.

a PyTorch tensor in the range [0.0, 1.0] (of shape 3×H×W for RGB). This conversion is needed since our network expects floating-point input.

For DETR and our baseline pipelines, we normalize our data by subtracting 0.5 from each channel and dividing by 0.5, mapping pixel values from [0, 1] into approximately [–1, +1] because it centers the data around zero and maps to approximately unit variance, which helps stabilize training.

For our DINO pipeline, we use Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]). This is as we use a pretrained model for our DINO implementation [4]. Thus, our DINO backbone is initialized with weights pretrained on ImageNet, which has the aforementioned mean and standard deviation. This alignment preserves zero-mean, unit-variance statistics in each channel, yielding stable feature-map responses and well-conditioned gradients. As a result, the network converges rapidly and reliably when fine-tuned on our astronomical images, rather than suffering from shifted activation distributions or gradient instability.

For the final step of pre-processing, we convert each image, label (for the MNIST dataset, each label is an `np.int64` scalar ranging from 0–3 corresponding to the class index), and bounding box into tensors, then pass our training, validation, and holdout sets into `torch.utils.data.DataLoader` to build compatible PyTorch datasets.

4.1.2 Augmentation

For training more advanced baseline tests and our final GZ2 experiment, we apply the following data augmentation transforms on the training set:

1. `transforms.RandomHorizontalFlip()` is applied to each batch.
2. Additionally, `transforms.RandomRotation()` is applied with a random rotation angle in ±10°.

For reference, Hausen and Robertson use additional vertical flips and random crops of size 60×60 and 40×40 [8]. However, as we show, our pipeline achieves better results with less augmentation, maximizing holdout accuracy. Most remaining misclassifications arise from bad or empty images and human labeling error. Additionally for the GZ2 Dataset, some images had significant disagreement between volunteers and so no classification was made. We filter out these datapoints.

4.1.3 Ground Truth Box Label Extraction for Hungarian Matching

The DETR pipeline is designed to incorporate both bounding box regression and class label prediction into its loss function. While ground truth class labels are provided in the dataset, 2D axis-aligned bounding boxes for the galaxies are not. To generate these bounding boxes, we follow a five-step process under the following assumption: the galaxy corresponding to the ground truth label (determined by a majority vote) is the largest non-noise or "foreground" object in each image frame across the dataset [8].

First, once the data is downloaded, each image is extracted and resized to 128×128 pixels (preserving RGB channels) using a transform [6]. Second, the images are converted to grayscale and represented as NumPy arrays [2]. Third, a binary mask is created for each image by thresholding pixel values above 50. Fourth, the largest connected component in the mask is identified using OpenCV's `connectedComponents` function. Finally, the bounding box coordinates are computed from the minimum and maximum x and y values of the detected component and normalized to the image size. The bounding box parameters are calculated using the following equations:

$$\begin{aligned}
center\_x &= \frac{(x\_min + x\_max)}{2.0 \cdot W} \\
center\_y &= \frac{(y\_min + y\_max)}{2.0 \cdot H} \\
width &= \frac{(x\_max - x\_min)}{W} \\
height &= \frac{(y\_max - y\_min)}{H}
\end{aligned}$$

These values are stored as the ground truth bounding box for each image and are used to supervise the DETR pipeline via Hungarian matching.

5. Experiments

For our experiments, we ran three main baseline tests on smaller binary and multiclass galaxy datasets, followed by one final test on the full Galaxy Zoo 2 dataset. The first two baselines focused on validating our pipeline's functionality; the final tests incorporated, more robust self-supervised feature extraction, holdout accuracy, and thorough hyperparameter optimization.

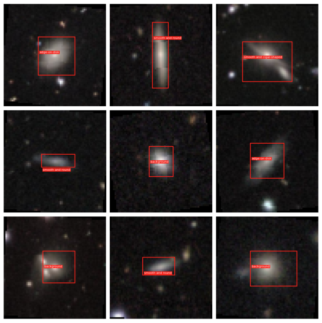

Figure 2. Example images from the MNIST dataset with bounding boxes and ground truth labels. Labels indicate the four classes.

5.1. Binary Classification on Ring Galaxies

To evaluate accuracy on the binary-class Demo Rings dataset, we consider only exact matches to the ground-truth labels. To prevent transformer queries from matching background or empty space, we added a "no-object" class alongside the "Ring" and "No Ring" classes. The Demo Rings set is relatively small (800 training, 200 validation), allowing aggressive tuning at the expense of compute time. Although each image contains one labeled object, we set the number of learnable queries to $N = 25$ so the model had ample capacity to refine. Since only one object appears per image, query competition is not a concern. We used a simple CNN feature extractor (2.1). For a dataset with only ring vs. non-ring galaxies, a shallow CNN sufficed, as confirmed by achieving 100% validation accuracy on Demo Rings.

For the CNN backbone of our DETR, one key hyperparameter was the output dimension D. We needed: 1) Enough feature channels for transformer queries to learn rich representations. 2) D to be divisible by the number of heads h in the multi-head attention module. For this baseline (and the first GalaxyMNIST test), we chose $D = 256$ to feed into an $h = 8$ head transformer. In practice, $D = 256$ provided sufficient capacity without excessive cost, and dividing evenly into 8 heads ensured each head received a 32-dimensional subspace.

On Demo Rings, the model achieved 100% validation accuracy, confirming that the simple CNN + DETR setup can perfectly classify ring vs. non-ring galaxies when only one object per image must be detected.

5.2. Multi-Class Baseline on GalaxyMNIST

Next, we tested on the GalaxyMNIST dataset [14], which contains ten times more images and four classes (all distinct from Demo Rings). We downloaded GalaxyMNIST and extracted features using the same CNN backbone. Validation accuracy dropped from 100% (Demo Rings) to 94% on GalaxyMNIST. As an initial error diagnostic, we visualized misclassified examples (5.5.1).

5.3. Hyperparameter Fine-Tuning for GalaxyMNIST

5.3.1 Pre-tuning Hyperparameters

To establish baseline performance, we began with rough estimates for key hyperparameters. To conserve memory, we initially set the batch size to 8, later increasing it to 32 once we confirmed that Colab's T4 GPU could handle the load without exceeding RAM limits.

For our learning rate, we started off with a smaller 1e-4 to work with our complex DETR pipeline and AdamW optimizer [10]. Additionally, we chose to run baseline tests with only 5 epochs, which our smaller starting batch size allowed for due to more frequent updates during training.

For our multi-head transformer model, we initialized the hidden dimension (d_{model}) to 256, the number of attention heads to 8, encoder and decoder layers to 3 each, and the number of queries to 25.

5.3.2 Tuned Hyperparameters

Through our MNIST baseline tests, we tuned hyperparameters for the final gz2 holdout tests. After upgrading to a Colab Pro session with an A100 GPU, we increased the batch size to 100. This allowed us to better leverage PyTorch's parallelization capabilities and train with faster epochs and improved GPU utilization. After baseline tests, using a larger batch size allowed us to increase epochs to 10, providing for additional full run throughs of the training data set.

We experimented with a higher learning rate in an attempt to accelerate convergence of the weights corresponding to minimizing the loss function over epochs. However, due to the complexity of the model, the DETR pipeline remained very sensitive to changes in learning rate, so we ultimately decided to stick with the 1e-4 learning rate.

We also increased the transformer hidden dimension to 512 for better feature expressivity and reduced the number of queries to 5, since the downstream task only required predicting the location and class of a single galaxy per image.

5.3.3 Learning Rate Scheduler

Additionally, we added a learning rate scheduler. Specifically, we used torch.optim's ReduceLROnPlateau scheduler with factor=0.5, patience=2, and threshold=1e-2 (see experiments for rationale).

5.3.4 DINO Backbone and Holdout Split

To improve feature learning, we replaced the CNN backbone with a self-supervised DINO backbone [3]. This choice immediately boosted validation accuracy by

2–3%. We also introduced a holdout set by reserving 10% of the GalaxyMNIST training data. To compensate for reduced training size, we increased augmentation by adding `RandomHorizontalFlip` and `RandomRotation`. With no learning-rate scheduler, the pre-optimized model (5.3.3) achieved 97.00% holdout accuracy.

5.3.5 Optimization and GalaxyMNIST Results

After hyperparameter tuning (5.3.2), the model reached 98.00% holdout accuracy, with an additional 0.5% gain using a validation accuracy plateau learning-rate scheduler. Early saturation of validation accuracy after one epoch indicated the efficacy of using DINO to find robust class feature representations. Training accuracy began at approximately 50% after the first epoch but quickly increased to match validation accuracy in subsequent epochs. This initial mismatch between training and validation performance likely stems from the model's difficulty in immediately learning stable feature representations from training data augmented with random transformations. Since these augmentations vary from batch to batch, they may introduce noise that temporarily impedes convergence. Once the model adapts to the augmented patterns, training accuracy rapidly catches up. To push past 98%, we added dropout (probability 0.1) after each encoder/decoder layer and reduced the learning rate on plateau. Dropout prevented overfitting, and the scheduler avoided oscillations from a too-high learning rate. The final 0.5% improvement was meaningful given only 2% room for progress.

5.3.6 Summarizing Baseline Experimental Results

- **CNN backbone, no DINO:** 94.00% validation accuracy.
- **DINO backbone, no scheduler:** 97.00% holdout accuracy.
- **DINO + tuning + scheduler:** 98.00% holdout accuracy (+0.5% with scheduler).

5.4. Large-Scale Multiclass Evaluation on Galaxy Zoo 2

The Galaxy Zoo 2 is the largest dataset we test on, consisting of nearly 300,000 galaxies. Notably, unlike the previous datasets, there are 7 classifications for galaxies. We trained the most successful model achieved on the MNIST dataset on this new dataset to verify that our methods are generalizable. This dataset is slightly harder given the larger number of samples and increased number of classes.

5.5. Results

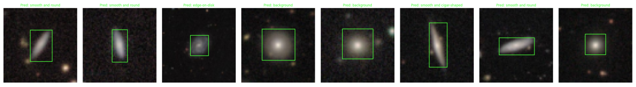

Figure 3. Bounding boxes learned for gz2 holdout data.

We see that using transformers-based object detection outperforms our ResNet-18 baseline when detecting galaxy morphology. Our DETR-based architectures improve the overall accuracy from our baseline metric by 23.50% (CNN backbone) and 25.30% (DINO backbone) respectively. Using the best model configuration achieved on MNIST (DINO backbone with Learning Rate scheduler, we achieve a classification accuracy of over 95% on the GZ2 dataset. Furthermore, our model is able to successfully learn precise bounding boxes for GZ2 holdout dataset 3.

Class / Metric	**ResNet-18**	**DETR + CNN**	**DETR + DINO**
smooth & round	0.85	1.00	0.99
smooth & cigar-shaped	0.72	0.98	0.99
edge-on-disk	0.58	0.98	0.99
unbarred-spiral	0.69	0.93	0.97
Overall Accuracy	72.45%	95.50%	97.75%
Macro-averaged F1	0.713	0.777	0.790
Weighted F1	0.713	0.972	0.988

Table 3. Comparison of per-class F1-scores and overall metrics across three models on the MNIST dataset.

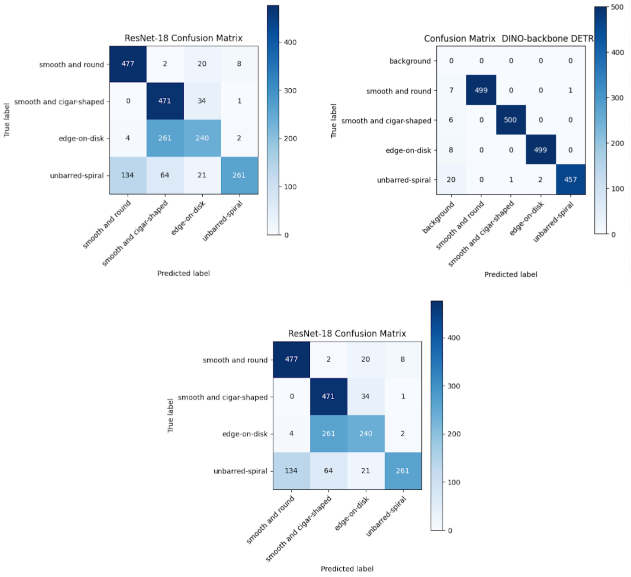

Figure 4. (Top Left) DETR + CNN validation confusion matrix (Top Right) DETR + DINO validation confusion matrix. (Bottom) ResNet-18 validation confusion matrix.

Metric	Flat LR (1e-4)		Scheduler	
	Val	Hold-out	Val	Hold-out
Accuracy (%)	97.60	98.00	98.45	98.50
Macro Precision	0.792	0.800	0.799	0.800
Macro Recall	0.780	0.784	0.787	0.788
Macro F_1	0.786	0.790	0.793	0.794
Weighted Precision	0.991	0.996	0.998	1.000
Weighted Recall	0.976	0.980	0.985	0.985
Weighted F_1	0.983	0.988	0.991	0.992

Table 4. DETR with DINO performance comparison on GalaxyMNIST: "Flat LR (1e-4)" vs. "ReduceLROnPlateau Scheduler." Metrics are reported for both the validation and hold-out splits.

Galaxy Zoo 2		
Metric	Val	Hold-out
Accuracy (%)	96.55%	95.36%
Macro Precision	0.873	0.841
Macro Recall	0.800	0.807
Macro F_1	0.832	0.810
Weighted Precision	0.998	0.973
Weighted Recall	0.965	0.954
Weighted F_1	0.981	0.959

Table 5. DETR with DINO performance on Galaxy Zoo 2

5.5.1 Qualitative Failure Mode Analysis

Many of the misclassified stamps in Figure X arise from extremely low signal-to-noise or partial occlusion, making it nearly impossible to discern the correct morphology even for a human observer. In several examples, the galaxy appears as a faint, isolated "dot" or is obscured by a bright foreground star or cosmic-ray artifact. When a stamp contains only a tiny, low-contrast core, the network cannot reliably extract features that distinguish, a "smooth & cigar-shaped" profile from a faint spiral arm. In these cases, the model frequently defaults to predicting a class like "unbarred-spiral" or "background," mirroring the uncertainty one would expect if a trained astronomer were to classify the same image.

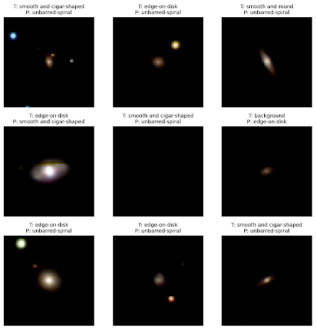

Figure 5. Misclassified images from the MNIST dataset from the DINO pipeline with predicted class and true class

Beyond pure image quality, some failure modes reflect intrinsic ambiguity at the class boundary. For example, an inclined disk with very faint arms can plausibly be labeled either "edge-on disk" or "unbarred spiral," and indeed the Galaxy Zoo volunteers themselves occasionally disagreed on these borderline examples. Given these observations, it is clear that (a) any realistic classification system must either filter out extremely low-signal stamps or assign them to an "uncertain" category, and (b) augmenting the training set with additional high-contrast examples (or using a higher-resolution telescope) would help reduce these ambiguities.

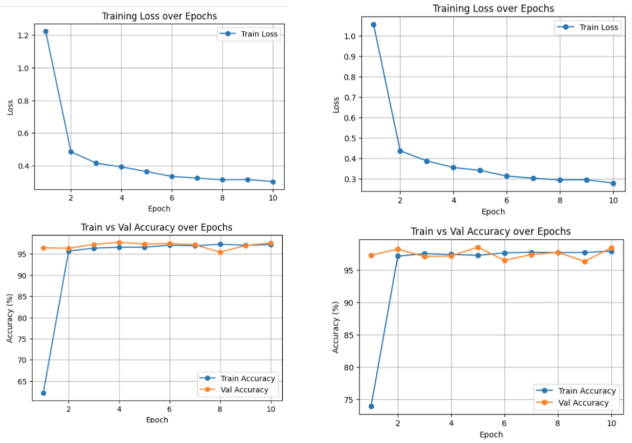

Figure 6. (Left) Training/validation loss curves without scheduler. (Right) Training/validation loss curves with ReduceLROnPlateau scheduler.

6. Conclusion

In conclusion, our works shows the promise of using DETR with a self supervised DINO feature extraction method for galaxy detection and classification. We first apply these methods to smaller datasets like Demo Rings and Galaxy MNIST, and finally apply our best model architecture to the Galaxy Zoo 2 dataset, where we achieve a classification accuracy of over 95%. Adapting our methods across different datasets shows the robustness of these results. Given that the crowd sourced datasets take years to assemble, methods like this could be used to achieve adequate summaries of new survey data immediately.

6.1. Future Works

Future work could incorporate additional data augmentation transforms such as random flilps, crops, and color jitter to the training set. Augmentation may allow for increased accuracy on novel, unlabeled datasets. [8].

Moreover, an automated data-filtering step (e.g., applying segmentation to each image and discarding those without any connected component above a size threshold) could remove poor-quality samples before training.

References

[1] M. Banerji, O. Lahav, C. J. Lintott, F. B. Abdalla, K. Schawinski, S. P. Bamford, D. Andreescu, P. Murray, M. J. Raddick, A. Slosar, and ... Galaxy zoo: Reproducing galaxy morphologies via machine learning. *Monthly Notices of the Royal Astronomical Society*, 406(1):342–353, 07 2010.

[2] P. Barchi, R. de Carvalho, R. Rosa, R. Sautter, M. Soares-Santos, B. Marques, E. Clua, T. Gonçalves, C. de Sá-Freitas, and T. Moura. Machine and deep learning applied to galaxy morphology - a comparative study. *Astronomy and Computing*, 30:100334, Jan. 2020.

[3] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko. End-to-end object detection with transformers, 2020.

[4] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. *arXiv preprint arXiv:2104.14294*, 2021.

[5] H. Domínguez Sánchez, M. Huertas-Company, M. Bernardi, D. Tuccillo, and J. L. Fischer. Improving galaxy morphologies for sdss with deep learning. *Monthly Notices of the Royal Astronomical Society*, 2018. Accepted for publication; arXiv:1711.05744 [astro-ph.GA].

[6] R. E. González, R. P. Muñoz, and C. A. Hernández. Galaxy detection and identification using deep learning and data augmentation, 2018.

[7] R. E. Hart, S. P. Bamford, K. W. Willett, K. L. Masters, C. Cardamone, C. J. Lintott, R. J. Mackay, R. C. Nichol, C. K. Rosslowe, B. D. Simmons, and ... Galaxy zoo: Comparing the demographics of spiral arm number and a new method for correcting redshift bias. *Monthly Notices of the Royal Astronomical Society*, 461(4):3663–3682, 10 2016.

[8] R. Hausen and B. E. Robertson. Morpheus: A deep learning framework for the pixel-level analysis of astronomical image data. *The Astrophysical Journal Supplement Series*, 248(1):20, May 2020.

[9] P. Jia, R. Sun, N. Li, Y. Song, R. Ning, H. Wei, and R. Luo. Detection of strongly lensed arcs in galaxy clusters with transformers. *The Astronomical Journal*, 165(1):26, Dec. 2022.

[10] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2017.

[11] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks, 2015. arXiv:1506.01497 [cs.CV].

[12] M. Walmsley, M. Bowles, A. M. Scaife, J. S. Makechemu, A. J. Gordon, A. M. Ferguson, R. G. Mann, J. Pearson, J. J. Popp, J. Bovy, J. Speagle, H. Dickinson, L. Fortson, T. Géron, S. Kruk, C. J. Lintott, K. Mantha, D. Mohan, D. O'Ryan, and I. V. Slijepevic. Scaling laws for galaxy images, 2024. arXiv:2404.02973 [cs.CV].

[13] K. W. Willett, C. J. Lintott, S. P. Bamford, K. L. Masters, B. D. Simmons, K. R. V. Casteels, E. M. Edmondson, L. F. Fortson, S. Kaviraj, W. C. Keel, T. Melvin, R. C. Nichol, M. J. Raddick, K. Schawinski, R. J. Simpson, R. A. Skibba, A. M. Smith, and D. Thomas. Galaxy zoo 2: detailed morphological classifications for 304 122 galaxies from the sloan digital sky survey. *Monthly Notices of the Royal Astronomical Society*, 435(4):2835–2860, Sept. 2013.

[14] W. Willett, Kyle and S. a. R. A. S. G. a. e. Domínguez Sánchez, Hugo de la Torre. Galaxy zoo decals: Detailed visual morphological classifications for dark energy camera legacy survey images of galaxies within the sdss dr8 footprint. *Monthly Notices of the Royal Astronomical Society*, 509(3):3966–3977, 2022.