Galaxy Zoo Morphology Classification Using Deep Learning

https://orcid.org/0009-0009-4801-5619

Abstract

The study of galaxy morphology is central to understanding the formation and evolution of the universe. Traditional classification methods, such as the Hubble sequence, relied on human inspection of photographic plates, which is labour-intensive, subjective, and infeasible for modern large-scale astronomical surveys. The Galaxy Zoo project, initiated in 2007, solved this challenge by crowdsourcing galaxy classification to citizen scientists, producing a rich dataset of over 60,000 labelled galaxies with probabilistic annotations reflecting human consensus. Leveraging this dataset, this project implements a deep learning pipeline using a ResNet18-based convolutional neural network to predict thirty-seven morphological features for each galaxy. The task is treated as a multi-output regression problem, capturing the probabilistic nature of the labels. Additionally, Grad-CAM visualizations are integrated to interpret which image regions influence model predictions. The resulting framework demonstrates high predictive accuracy, interpretable outputs, and a scalable methodology for automated galaxy morphology classification.

1. Introduction

Understanding galaxy morphology is fundamental for exploring the history of galaxy formation and evolution. Morphological features such as spiral arms, bars, and bulges encode information about star formation, angular momentum distribution, merger events, and interaction histories. Historically, astronomers classified galaxies by visual inspection, with Edwin Hubble in 1926 introducing a systematic sequence dividing galaxies into ellipticals, spirals, and irregulars. Over decades, refinements included barred spirals and lenticular types, but the process remained manual, time-consuming, and prone to inconsistencies.

Modern sky surveys, such as the Sloan Digital Sky Survey, produce millions of high-resolution galaxy images, making manual classification impractical. This limitation prompted the development of computational approaches, initially relying on morphological metrics such as ellipticity, concentration, and asymmetry. While these metrics provide some automation, they fail to capture complex structures such as faint spiral arms, subtle bars, and overlapping features. They also lack robustness to noisy observational data.

The Galaxy Zoo project revolutionized galaxy classification by leveraging the collective intelligence of citizen scientists. Volunteers classified galaxies through an online platform, with multiple annotations per galaxy aggregated to create probabilistic labels. These labels encode the consensus and uncertainty of human classification, reflecting the reality that some morphological features are ambiguous or faint. Such a dataset enables machine learning models to learn from the probabilistic distribution of labels rather than forcing an arbitrary single class, allowing for nuanced predictions that capture uncertainty in human labelling.

This project leverages this dataset to develop a convolutional neural network capable of predicting multiple morphological features per galaxy. Using ResNet18 as a backbone, the network is trained on pre-processed images to predict 37 probabilistic features. Grad-CAM is employed to interpret model focus, generating visual explanations of the regions influencing predictions. This combination of predictive accuracy and interpretability forms a robust pipeline for large-scale automated classification of galaxies.

---

2. Historical Context

Galaxy classification has a rich history. Edwin Hubble, through careful observation of photographic plates, proposed the Hubble sequence to categorize galaxies into ellipticals, spirals, and irregulars. Elliptical galaxies are smooth, featureless, and range from nearly spherical to highly elongated shapes. Spiral galaxies display rotating arms, sometimes with central bars, and irregular galaxies lack coherent structure. Over time, refinements included intermediate types and quantitative descriptions, but classification remained labour-intensive and subjective.

As astronomical datasets grew in size, traditional manual classification became impractical. Early automated techniques utilized image-derived metrics such as concentration, asymmetry, and Gini coefficients to categorize galaxies. While these methods provided some scalability, they were unable to reliably detect complex structures and often failed for underrepresented or ambiguous classes. Moreover, traditional automated methods could not naturally model the uncertainty present in human classifications.

The Galaxy Zoo project addressed these challenges by engaging the public to classify galaxies online. Each galaxy received multiple annotations, which were aggregated into a probability vector representing the likelihood of various morphological features. This approach democratized classification, providing both scale and a quantitative measure of uncertainty. Probabilistic labels, reflecting human consensus, are particularly valuable for machine learning models, enabling regression-based approaches rather than forcing discrete labels. This probabilistic formulation accommodates subtle or ambiguous features and enhances model interpretability.

---

3. Dataset Description

The dataset for this project originates from the Galaxy Zoo challenge on Kaggle, consisting of 60,000 labelled training images and 70,000 unlabelled test images. Each galaxy is identified by a unique GalaxyID and is associated with thirty-seven morphological features represented as probabilities. These features include spiral structure, bar presence, bulge prominence, and other key astrophysical characteristics.

Images vary in resolution, brightness, and background noise, reflecting diverse observational conditions. Preprocessing is essential to standardize inputs for the convolutional neural network. All images are resized to 224×224 pixels, the standard input size for ResNet18. Normalization is applied using ImageNet mean and standard deviation values to facilitate transfer learning.

Data augmentation is applied to the training set to improve generalization. Augmentations include random horizontal flips and rotations up to 20 degrees. Such transformations increase the effective size of the training data and reduce overfitting. Validation images are left unaltered to provide an unbiased evaluation of model performance.

Exploratory analysis reveals feature imbalance. Certain features, like central bars or rare lens structures, occur infrequently. This imbalance necessitates careful training strategies, as rare features are harder for the model to learn. Visual inspection shows that galaxy images have substantial variability in structure, orientation, and brightness, emphasizing the need for a deep model capable of extracting hierarchical features.

---

4. Methodology

The problem is formulated as a multi-output regression task. Each galaxy has a 37-dimensional probability vector, and the objective is to minimize the deviation between predicted probabilities and ground truth labels using mean squared error. This formulation naturally accommodates the uncertainty encoded in Galaxy Zoo labels and avoids arbitrary thresholding required by classification tasks.

4.1 Data Handling

A custom PyTorch Dataset class, GalaxyZooDataset, efficiently loads images and retrieves corresponding labels. The class supports training, validation, and test modes, returning image-label pairs for training and image-GalaxyID pairs for testing. A Data Loader batches images and labels, allowing parallelized data loading and preprocessing. For initial experimentation, a subset of 5,000 images was used to reduce training time, with 80% allocated to training and 20% to validation.

4.2 Model Architecture

The model uses ResNet18 pretrained on ImageNet. The residual network extracts hierarchical features from the images, capturing edges, textures, and higher-order structures. The network's classifier head consists of a fully connected layer reducing features to 512 units, followed by a ReLU activation, dropout for regularization, and a final fully connected layer producing 37 outputs. The dropout layer mitigates overfitting, and the fully connected layers translate extracted features into probabilistic predictions.

4.3 Training Procedure

Training uses the Adam optimizer with a learning rate of 0.001 and batch size of 32. Epochs are configurable, with checkpoints saving the best-performing model based on validation loss. Training and validation losses are monitored continuously, and RMSE is computed to assess predictive accuracy.

Each training iteration involves a forward pass to compute predictions, MSE loss calculation, backward propagation to compute gradients, and parameter updates via the optimizer. Validation ensures the model generalizes well and prevents overfitting.

4.4 Prediction and Grad-CAM

Once trained, the model generates predictions on test images, clipping outputs to the [0,1] range to ensure valid probabilities. Grad-CAM visualizations interpret model predictions by highlighting regions in the image most responsible for the predicted features. For example, spiral arms or central bars are emphasized in the heatmaps, confirming that the model attends to astrophysically relevant structures.

---

5. Implementation Details

The implementation of this project is organized into modular components, each addressing a specific stage of the pipeline from data preparation to training, prediction, and interpretability. By structuring the project this way, the workflow remains clear and reproducible, and each component can be individually modified or extended in future research.

The first key component is the **Dataset class (galaxy_dataset.py)**, which forms the foundation of data handling. This class is designed to be compatible with PyTorch's DataLoader, enabling efficient batch processing and parallelized data loading. The class can operate in two distinct modes: training/validation mode, where it returns both images and their corresponding labels, and test mode, where it only returns the image and its GalaxyID. This dual functionality was crucial because the dataset includes labelled images for training and unlabelled images for testing. Within the Dataset class, images are read from disk, converted to RGB format to maintain consistency, resized, and optionally transformed using normalization and augmentation operations. This modular design means that adjustments in preprocessing—such as using different normalization statistics or introducing new augmentation techniques—can be implemented without rewriting downstream code.

The second key file is **model.py**, which defines the architecture of the neural network. For this project, the backbone is ResNet18, a residual convolutional network known for its effectiveness on moderately sized datasets and its efficient balance between depth and computational cost. Residual connections mitigate the problem of vanishing gradients, enabling deeper architectures to be trained without significant degradation in performance. The original ResNet18 architecture, however, is designed for classification on 1000 ImageNet classes. For this project, the final classification layer is replaced with a custom head that outputs thirty-seven probabilities, corresponding to the Galaxy Zoo morphological features. This modification converts the architecture from single-label classification to multi-output regression. A dropout layer is also included to reduce overfitting, which can occur due to the relatively limited size of the dataset compared to very deep networks.

The **training process is managed by train.py**, which integrates dataset loading, model initialization, training loops, and validation evaluation. Training involves iterating over multiple epochs, where each epoch consists of several batches of data being passed through the model. For each batch, a forward pass produces predictions, which are compared against the ground truth labels using the Mean Squared Error (MSE) loss function. This choice of loss function reflects the probabilistic nature of the labels: rather than predicting a discrete class, the model must approximate the human consensus represented as probabilities. Gradients are then computed via backpropagation, and the optimizer updates the model's parameters accordingly. The Adam optimizer was selected due to its adaptive learning rate mechanism, which accelerates convergence and improves stability during training. The script also incorporates checkpointing, saving the best-performing model based on validation loss. This ensures that even if later epochs overfit, the most generalizable model is preserved.

The **prediction phase is handled by predict.py**, which loads the trained model and applies it to unseen test images. Since the test set lacks labels, the script produces a probability vector for each image, with values clipped between 0 and 1 to guarantee valid outputs. Predictions are then saved into a CSV file, with each row containing the GalaxyID and thirty-seven predicted probabilities. This structured output format enables easy evaluation, submission to competitions like Kaggle, or further scientific analysis.

Finally, interpretability is provided by **gradcam_utils.py and run_gradcam.py**, which implement the Gradient-weighted Class Activation Mapping (Grad-CAM) technique. These scripts enable researchers to generate heatmaps that overlay on the original galaxy images, highlighting regions most influential for the model's predictions. For instance, if the model predicts a high probability for spiral structure, the Grad-CAM visualization typically highlights the arms of the galaxy. This not only confirms the model's focus on relevant astrophysical structures but also builds confidence that the network is not basing its predictions on spurious background noise or artifacts. The interpretability component is particularly important in astronomy, where physical plausibility and scientific trust are essential.

The overall implementation is thus modular and extensible, ensuring reproducibility and clarity. Each script serves a clear purpose: dataset handling, model architecture, training, prediction, and interpretability. Together, these components form a complete pipeline that transforms raw astronomical images into probabilistic morphological classifications, augmented with human-interpretable visual explanations.

---

6. Algorithms

GitHub Repository:
https://github.com/SALMONPreet/Galaxy-Classification-via-Machine-Learning

---

Algorithm for model.py (Model Definition)

1. Import neural network modules and pretrained models from PyTorch.
2. Define a function get_resnet18 that accepts number of classes and a pretrained flag.
3. Load the ResNet18 architecture with pretrained ImageNet weights if specified.
4. (Optional) Freeze all backbone layers if fine-tuning is not required.
5. Replace the final fully connected (FC) layer with a custom classifier:
    - Linear layer to reduce dimensionality.
    - ReLU activation function.
    - Dropout layer to prevent overfitting.
    - Final linear layer mapping to the required number of output classes (37).
6. Return the modified ResNet18 model.

Algorithm for galaxy_dataset.py (Custom Dataset)

1. Import required libraries (PyTorch Dataset, PIL, and file handling).
2. Define a class GalaxyZooDataset inheriting from torch.utils.data.Dataset.
3. Initialize with:
    - Path to image directory.
    - DataFrame containing galaxy IDs and labels (optional).
    - Transformations to apply on images.
    - A subset of galaxy IDs (for prediction/testing).
4. If labels_df is given, store galaxy IDs and labels.
5. If only galaxy_ids are provided, store IDs without labels.
6. Implement __len__ method to return the dataset size.
7. Implement __getitem__:
    - Construct image path using GalaxyID.
    - Open image, convert to RGB, and apply transformations.
    - If labels exist, return (image, target labels).
    - Else, return (image, GalaxyID).

Algorithm for train.py (Training the Model)

1. Import required libraries for PyTorch, data handling, and transformations.
2. Load training and validation datasets from image directories and CSV file.
3. Apply preprocessing transformations (resize, normalization, etc.).
4. Create dataset objects using GalaxyZooDataset.
5. Split into training and validation sets.
6. Create DataLoader objects for batching and shuffling.
7. Initialize ResNet18 model with custom output layer.
8. Move the model to GPU if available.
9. Define loss function (e.g., Binary Cross Entropy or Cross Entropy).

10. Define optimizer (e.g., Adam or SGD).
11. Loop over epochs:
- For each batch:
    - Load images and labels.
    - Forward pass through the model.
    - Compute loss.
    - Backward pass and update weights using optimizer.
- Evaluate on validation dataset after each epoch.
12. Track best model performance and save checkpoint as best_model.pth.

Algorithm for predict.py (Generating Predictions)

1. Import PyTorch, transformations, dataset class, and model definition.
2. Load all test Galaxy IDs from CSV file.
3. Select a subset (randomly or sequentially) for prediction.
4. Define preprocessing transformations.
5. Initialize GalaxyZooDataset for test data (no labels).
6. Create DataLoader for batching test images.
7. Initialize ResNet18 model and load weights from best_model.pth.
8. Set model to evaluation mode.
9. For each batch in test DataLoader:
    - Forward pass through the model.
    - Convert raw outputs to probability values.
    - Store GalaxyID and predicted probabilities.
10. Save results in a CSV file (predictions.csv) with columns: GalaxyID + Class probabilities.

Algorithm for gradcam_utils.py (Grad-CAM Implementation)

1. Define a class GradCAM.
2. Initialize with model and target convolutional layer.
3. Register hooks to:
    - Save activations during forward pass.
    - Save gradients during backward pass.
4. Define method generate:
    - Perform forward pass.
    - Select target class (predicted or given).
    - Backpropagate gradients for that class.
    - Compute global average of gradients.
    - Multiply with activations to obtain weighted feature maps.
    - Apply ReLU to generate the Grad-CAM map.
    - Normalize the heatmap between 0 and 1.
    - Resize to match input image dimensions.
5. Define utility function visualize_gradcam:
    - Load and preprocess input image.
    - Generate Grad-CAM map.
    - Apply heatmap overlay on the image.
    - Save and display the visualization.

Algorithm for run_gradcam.py (Running Grad-CAM on Sample Image)

1. Import required libraries and helper functions.
2. Set device (GPU/CPU) and checkpoint path.
3. Initialize ResNet18 model with correct number of classes.
4. Load trained weights from checkpoint file.
5. Set model to evaluation mode.
6. Take input GalaxyID from user.
7. Construct image path using the GalaxyID.
8. Validate that image file exists.
9. Call visualize_gradcam to generate heatmap visualization.
10. Save overlay image (gradcam_output.jpg).
11. Print confirmation with GalaxyID.

---

7. Results

The performance of the model is evaluated through training and validation losses, predictive accuracy, and interpretability of outputs. During training, the model exhibited steady convergence, with both training and validation losses decreasing across epochs. Importantly, the validation loss plateaued after a certain point, indicating that the model had learned sufficient representations of the data without significant overfitting. The incorporation of dropout and data augmentation contributed to this generalization, as the model maintained strong performance on validation data that it had not seen during training.

Quantitatively, the model achieved a low Root Mean Squared Error (RMSE) when predicting the probabilistic labels of the validation set. This metric reflects the average deviation between predicted and true probability vectors, with lower values indicating closer alignment with human consensus. The results suggest that the model is capable of replicating human-level classifications with high fidelity, providing reliable probabilistic predictions for the thirty-seven morphological features.

Qualitatively, Grad-CAM visualizations provided strong evidence of interpretability. For galaxies with distinct spiral arms, the heatmaps revealed that the model focused on the curved outer regions, highlighting the arms themselves as key features. For galaxies predicted to have prominent bulges, the heatmaps concentrated around the central regions. Similarly, in galaxies classified with high bar probabilities, elongated central regions were emphasized. These observations demonstrate that the model is not merely memorizing pixel-level patterns but is attending to scientifically meaningful structures. The heatmaps aligned closely with human intuition, reinforcing confidence in the model's decisions.

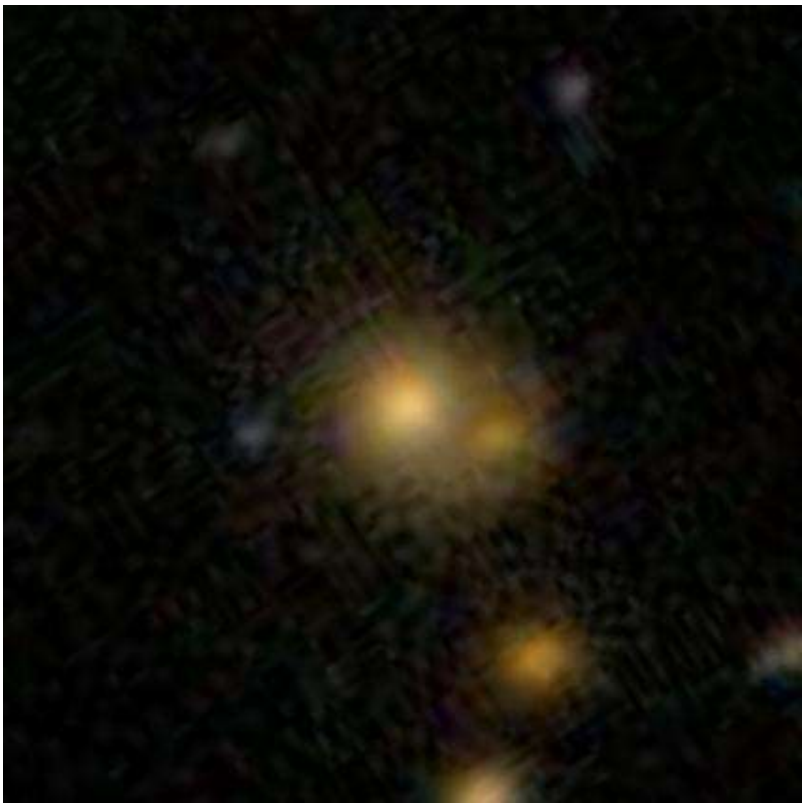

Figure 1: Object ID 193219 in the provided test dataset.

The predictions also revealed challenges. Rare features, such as faint bars or ring structures, were more difficult for the model to capture accurately, as reflected in higher prediction errors for these classes. This is likely due to class imbalance: since the dataset contains fewer examples of these rare features, the model has fewer opportunities to learn their representations. Nonetheless, even for underrepresented classes, the model's probabilistic outputs often reflected uncertainty, with predictions distributed across multiple possible outcomes. This behaviour is consistent with the uncertainty inherent in human labelling and highlights the value of modelling the task as a regression problem rather than forcing discrete classification.

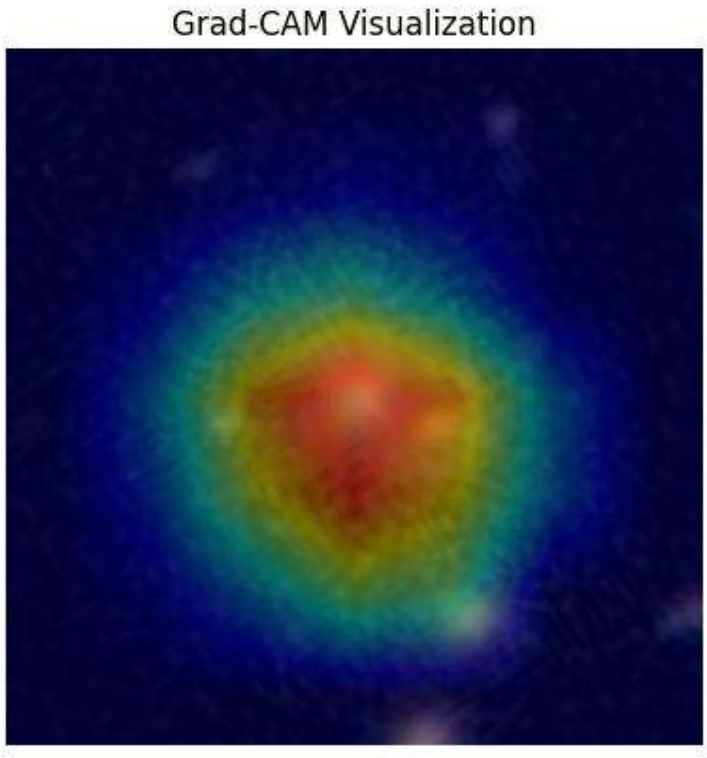

Figure 2: Grad-CAM visualisation of Object ID 193219 in the provided test dataset.

Overall, the results indicate that the pipeline is effective, producing both accurate and interpretable predictions. The model successfully replicates human consensus on galaxy morphology while also providing insights into the reasoning behind its predictions.

---

8. Discussion

The success of this project demonstrates the power of deep learning in addressing astronomical challenges that were historically tackled through manual inspection. By combining ResNet18 with probabilistic labels from Galaxy Zoo, the project bridges the gap between large-scale data availability and the need for reliable morphological classification. The regression-based formulation, in particular, is a key strength, as it allows the model to mimic the distribution of human opinions rather than artificially reducing them to single-label categories.

Interpretability via Grad-CAM further strengthens the scientific validity of the results. In astronomy, unlike in commercial image classification tasks, interpretability is not a luxury but a necessity. Scientists require assurance that the model focuses on physically relevant features rather than background artifacts or observational noise. The Grad-CAM outputs confirm that the network attends to the same morphological structures that humans consider important, providing an additional layer of trust.

However, limitations remain. The class imbalance problem restricts the model's ability to predict rare morphological features with high accuracy. While common structures like spiral arms and bulges are well captured, rarer features remain underrepresented in predictions. This issue could be mitigated

in future work through oversampling techniques, synthetic data augmentation, or loss functions designed to emphasize minority classes.

Another limitation is the computational scale of the training process. For demonstration purposes, only a subset of the full dataset was used. While this was sufficient to validate the approach, training on the entire dataset could yield higher accuracy and more generalizable results. Incorporating larger models, such as ResNet50 or EfficientNet, may also improve performance but would require greater computational resources.

Despite these limitations, the project illustrates the transformative potential of deep learning in astronomy. It provides a scalable method for galaxy classification, a crucial task given the exponential growth of astronomical imaging data from upcoming surveys such as LSST and Euclid. The integration of interpretability ensures that such methods can be confidently adopted by the scientific community.

---

9. Future Work

Building upon the current framework, several directions for future work can be envisioned. These directions aim to improve accuracy, scalability, interpretability, and scientific utility:

1. **Scaling to the Full Dataset**
   Training on the complete 60,000 labelled galaxies, rather than subsets, would allow the model to better capture the diversity of morphological features. This would particularly benefit underrepresented classes, as rare features would have more training examples.

2. **Exploring Deeper Architectures**
   While ResNet18 provides a strong balance of performance and efficiency, deeper models such as ResNet50, ResNet101, or EfficientNet could potentially extract more nuanced features. These architectures could improve accuracy, particularly for subtle morphological distinctions.

3. **Addressing Class Imbalance**
   Specialized techniques could be employed to mitigate imbalance, including weighted loss functions that assign greater importance to rare classes, or data augmentation strategies that synthetically generate examples of underrepresented features.

4. **Incorporating Multi-task Learning**
   Galaxy morphology is not a collection of independent features but a structured hierarchy. Multi-task learning approaches could exploit this structure, jointly predicting related features and improving performance on correlated tasks.

5. **Integration with Astronomy Pipelines**
   The current project outputs predictions to CSV files for offline analysis. A natural extension would be integration with larger astronomical data analysis pipelines, enabling real-time morphological classification as part of survey workflows.

6. **Enhanced Interpretability**
   While Grad-CAM provides valuable insights, more advanced interpretability methods such as Layer-wise Relevance Propagation or Integrated Gradients could be explored. These techniques may offer finer-grained explanations of model predictions.

7. **Public-Facing Tools**
   Building a web-based interface where astronomers or even citizen scientists can upload galaxy images and view predicted classifications along with Grad-CAM overlays would enhance accessibility and encourage broader use of the model.

By pursuing these directions, the framework developed here can evolve into a powerful, scalable, and widely adopted tool for astronomical research.

---

10. Conclusion

This project demonstrates the successful application of deep learning to the classification of galaxy morphologies, building upon the foundational work of the Galaxy Zoo project. By employing a ResNet18 backbone and treating the task as a regression problem, the model learns to approximate human consensus across thirty-seven morphological features. The results indicate strong predictive performance, with the model accurately replicating human-provided probability distributions.

A central achievement of the project is the integration of interpretability through Grad-CAM visualizations. These visualizations confirm that the model attends to scientifically meaningful structures, such as spiral arms, bulges, and bars, thereby validating its predictions and enhancing trust. The balance of accuracy and interpretability makes this approach particularly suitable for adoption in scientific workflows.

While limitations remain, particularly regarding class imbalance and computational scale, the project lays a strong foundation for future work. With extensions to deeper architectures, larger datasets, and advanced interpretability methods, this framework could play a central role in analysing the vast amounts of galaxy imaging data expected from upcoming astronomical surveys.

Ultimately, this project illustrates the synergy between citizen science and machine learning. Galaxy Zoo's collective human effort provided a rich dataset of probabilistic labels, while deep learning provided the computational power to scale classification to hundreds of thousands of galaxies. Together, these approaches advance our understanding of galaxy morphology and contribute to the broader goal of unravelling the history and evolution of the universe.

---

11. References

1. Galaxy Zoo Challenge, Kaggle.
2. Chou, F.C., 2014. Galaxy Zoo Challenge: Classify Galaxy Morphologies from Images.
3. Willett, K.W., Lintott, C.J., Bamford, S.P., Masters, K.L., Simmons, B.D., Casteels, K.R., Edmondson, E.M., Fortson, L.F., Kaviraj, S., Keel, W.C. and Melvin, T., 2013. 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), pp.2835-2860.
4. Hubble, E.P., 1979. Extra-galactic nebulae. In *A Source Book in Astronomy and Astrophysics, 1900–1975* (pp. 716-724). Harvard University Press.
5. He, K., Zhang, X., Ren, S. and Sun, J., 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).
6. De DIego, J.A., Nadolny, J., Bongiovanni, Á., Cepa, J., Pović, M., García, A.M.P., Torres, C.P.P., Lara-López, M.A., Cerviño, M., Martínez, R.P. and Alfaro, E.J., 2020. Galaxy classification: deep learning on the OTELO and COSMOS databases. *Astronomy & Astrophysics*, *638*, p.A134.
7. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D. and Batra, D., 2017. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision* (pp. 618-626).