Research in Astronomy and Astrophysics, 25:065018 (20pp), 2025 June
https://doi.org/10.1088/1674-4527/adce8f
CSTR: 32081.14.RAA.adce8f

An Interpretable Galaxy Morphology Classification Approach Using Modified SqueezeNet and Local Interpretable Model-agnostic Explanation

Kam Meng Goh[1,5,*], Derrick Hiang Yaol Lim[2,5], Zhen Dong Sham[3], and Kolla Bhanu Prakash[4]
[1] Centre for Multimodal Signal Processing, Faculty of Engineering and Technology, Tunku Abdul Rahman University of Management and Technology, 53300 Kuala Lumpur, Malaysia; gohkm@tarc.edu.my
[2] Starlight Education Group, London, W1W 5PF, UK
[3] Faculty of Institute for Advanced Studies, UM Power Energy Dedicated Advanced Centre, Universiti Malaya, 59990 Kuala Lumpur, Malaysia
[4] Department of Computer Science & Engineering, DST—FIST Sponsored Department, Koneru Lakshmaiah Education Foundation, Andhra Pradesh, 522502, India
Received 2024 September 11; revised 2025 March 23; accepted 2025 April 7; published 2025 June 3

Abstract

The recent surge in computer vision and deep learning has attracted significant attention within the galaxy morphology community. Various models have been implemented for galaxy morphology prediction with near-perfect accuracy for certain classes. However, many studies treat deep learning models as black-box entities, lacking interpretability of their predictions. To address these limitations while ensuring good performance, we introduced an Improved SqueezeNet (I-SqueezeNet) by incorporating unique residual connections to improve the prediction performance, and we utilize Local Interpretable Model-Agnostic Explanations (LIME) to understand the interpretability. We evaluated the simplified SqueezeNet and I-SqueezeNet, with both channel and vertical concatenation, and compared their performances with those of some exiting methods such as Dieleman's CNN, VGG13, DenseNet121, ResNet50, ResNext50, ResNext101, DSCNN and customized CNN in classifying galaxy objects using a dataset from the publicly available Galaxy Zoo Data Challenge Project. Our experiments showed that I-SqueezeNet with vertical concatenation achieved the highest average accuracy of 94.08% compared to other methods. Beyond achieving high accuracy, the application of LIME for model interpretation sheds light on the machine learning features and reasoning processes behind the model's predictions. This information provides valuable insight into the galaxy morphology decision-making process, paving the way for further functional enhancements.

Key words: methods: data analysis – methods: analytical – methods: statistical – techniques: image processing

1. Introduction

The classification of galaxies allows us to understand the structure and evolution of our universe. The morphological classification of galaxies was pioneered by Edwin Hubble (Hubble 1926) where he categorized galaxies according to their appearance on photographic plates, distinguishing galaxies into ellipticals (E0-E7), spirals (Sa-Sc), barred-spirals (SBa-SBc) and irregular galaxies (Irr I and Irr II). The morphology of galaxies infers the evolution and formation timeline of galaxies, and is correlated to a galaxy's stellar population, mass distribution, and dynamics. Additionally, the distribution of the different types of galaxies infers the large-scale structure of our universe. Efforts in galactic morphology identification started with visual inspection of photographic plates (Hubble 1926; De Vaucouleurs 1959; Sandage 1961; Fukugita et al. 2007; Nair & Abraham 2010; Baillard et al. 2011; Buta 2013) and has been the main approach for many decades. With the proliferation of astronomical imaging and significant advancements in observational instrumentation and technology, the amount and complexity of astronomical data has increased exponentially. With a multitude of sky surveys such as the Sloan Digital Sky Survey that may capture up to hundreds of millions of galaxy images, the sheer amount of computing power required renders traditional CPU-based algorithms (Abraham et al. 2003; Lotz et al. 2004) inefficient.

The classification of galaxy morphology through machine learning has drawn considerable attention in recent years, due to the efficacy of artificial intelligence in identifying and classifying celestial objects. Previous research demonstrated that machine learning has become a useful tool in galaxy-related research over the past few decades (Pasquet et al. 2018; Vavilova et al. 2021; Khramtsov et al. 2022). In most cases, the machine learning method consists of two fundamental steps: feature extraction and machine learning classification. Feature extraction is essential to identify key attributes from the provided data (i.e., images or statistical information), while machine learning classification involves learning and classifying these extracted attributes. Conventionally, feature extraction is a tedious process, since this process requires individuals

* Corresponding Author.
[5] Both authors contributed equal amounts of work in this paper.

to design mathematical models manually to extract specific attributes before classification. The pipeline of machine learning classification using handcrafted features (also known as low-level approaches) has been employed by some previous work. For instance, Naim et al. (1995) used morphological features and a supervised artificial neural network (ANN) to classify automated plate measuring machine (APM) galaxy images. Calleja & Fuentes (2004) adopted image analysis, principle component analysis, and several classifiers to classify galaxy images into three classes. Reza (2021), on the other hand, extracted the 25 most significant principal components from 62 features to be inputs for five different classifiers to perform galaxy morphology. Apart from these works, Banerji et al. (2010) used neural networks to classify galaxy images into three classes based on 12 *i*-band parameters. Instead of using *i*-band parameters, *r*-band photometric parameters and spectral parameters were also proposed as features to train the classification model based on the Classification and Regression Tree (CART), the C4.5 decision tree learners, the Random Forest, as well as fuzzy logic algorithms (Gauci et al. 2010). Sreejith et al. (2017) extracted 10 features from size, color, shape parameter, and stellar mass, and applied Support Vector Machines (SVM), Classification Trees (CT), Classification Trees with Random Forest and Neural Networks for classification. Coincidentally, Barchi et al. (2017) also used SVM and decision tree to improve the galaxy morphology classification process based on five morphology parameters. In short, all the listed methods here are considered as low-level approaches, since handcrafted features have been used as the key attribute to represent galaxy images.

Unlike low-level approaches, the deep learning approach is attracting increased attention for its ability to automatically learn and extract features. For instance, deep learning has demonstrated remarkable performance in the domain of image classification or object detection (Zhao et al. 2017). Previous literature has applied convolutional neural networks (CNNs) as a deep learning architecture in galaxy morphology classification. For example, Dieleman et al. (2015) has proven that a CNN is capable of recognizing galaxy images, where the authors claimed to have near-perfect accuracy. On the other hand, Zhu et al. (2019) proposed a variant of the Residual Network (ResNet) and compared the performance with that of Dieleman et al. (2015), as well as other models such as AlexNet, Inception, ResNet, and VGG for galaxy morphology classification, with a result of 95.21% accuracy. Barchi et al. (2020) proposed and compared their own CNN network with conventional low-level approaches in galaxy morphology recognition, and the results showed that deep learning can significantly outperform conventional approaches. However, their method also showed a decrement in performance if the number of classes increased. Incidentally, Cheng et al. (2020) also analyzed and compared the performance of a CNN with several common classifiers (i.e., SVM, random forest, etc.), and the CNN outperformed the other classifiers, with 98.7% accuracy under two class classification. Despite having the aforementioned CNN, some other approaches can be found in previous works for galaxy morphology classification, such as EfficientNet (Fielding et al. 2021; Kalvankar et al. 2020; Wu et al. 2022), or Capsule Net (CapsNet) with a reduction of 36.5% error rate in two class classification (Katebi et al. 2019). Additionally, DenseNet was also proposed for galaxy morphology classification based on the Galaxy Zoo DECals dataset with 84.0% F1-score (Fielding et al. 2021). Also, Dilated-Separable CNN (DSCNN) was proposed to classify galaxy objects based on the Galaxy Zoo 2 dataset with 89.74% accuracy for five-class classification (Waghumbare et al. 2024). The upgraded versions of ResNet, namely ResNext50 and ResNext101, were also found in the previous works to perform galaxy morphology classification in Galaxy 10 DECals dataset and Galaxy Zoo 2 dataset respectively, with at least 80% performance for ten-class classification and five-class classification respectively (Yu 2023; Jeevan & Sethi 2024). Recently, Urechiatu et al. customized a CNN to classify galaxy objects from the Galaxy Zoo 2 dataset, with 96.83% accuracy for five-class classification (Urechiatu & Frincu 2024). All the listed literature demonstrated that CNN is commonly used for galaxy morphology classification in the standard Galaxy Zoo dataset as a benchmark, since Galaxy Zoo is well established with a sufficient number of data to train the models. Nevertheless, it is widely recognized in the research community that deep learning typically requires extensive amounts of data. Hence, to address this issue, few-shot learning techniques like the Siamese network were introduced for galaxy morphology recognition (Zhang et al. 2022). These methods ensure reliable performance and, in some cases, even outperform common CNN approaches. Interestingly, machine learning and deep learning are not only employed for galaxy object classification, but have also been deployed for other kinds of astroinformatic study, such as stellar evolution track predictions (Maltsev et al. 2023) or astronomical text analysis (Shapurian 2024).

Despite promising results from deep learning, CNNs are frequently seen as black boxes, with the essential attributes or features that determine the predictions remaining unknown. Even though Dieleman et al. (2015) uncovered all the resulting convolutional filters in their proposed network after the training process, it becomes a challenging task to retrieve the trained filters for models with deep architecture. Previous work pointed out that using the decision made by machine learning models without proper justification or detailed explanation can be risky (Gunning et al. 2019). As a result, explainable artificial intelligence (XAI) has emerged as a trend in the machine learning community. The goal of XAI is to evaluate how a machine learning model makes predictions and decisions. The interpretability it provides ensures that the model's robustness is founded on significant and relevant features. In summary, XAI serves as a qualitative assessment tool to evaluate the

Table 1
Dataset Details

Annotation	Number of Samples	Threshold
Completely round smooth (class 0)	8436	$f_{\text{smooth}} \geqslant 0.469$
In-between smooth (class 1)	8069	$f_{\text{completelyround}} \geqslant 0.50$ $f_{\text{smooth}} \geqslant 0.469$ $f_{\text{inbetween}} \geqslant 0.50$
Cigar-shaped smooth (class 2)	579	$f_{\text{cigarshape}} \geqslant 0.50$ $f_{\text{smooth}} \geqslant 0.469$
Edge-on (class 3)	3903	$f_{\text{featuredisk}} \geqslant 0.430$ $f_{\text{edgeon,yes}} \geqslant 0.602$
Spiral (class 4)	7806	$f_{\text{edgeon,no}} \geqslant 0.715$ $f_{\text{spiral,yes}} \geqslant 0.619$

performance of a machine learning model, rather than acting as a feedback mechanism for performance improvements. The recent rise of XAI has made it applicable in various domains, such as healthcare (S Band et al. 2023) and finance (Weber et al. 2024). In the context of galaxy object classification, saliency mapping methods such as SmoothGrad (Bhambra et al. 2022) were proposed in the previous work to explain how deep learning performs galaxy morphology classification. However, on occasion, these approaches erroneously pointed out areas that models did not genuinely utilize to predict the outcome (Draelos & Carin 2021). Currently, there are very few research efforts that focus on emphasizing the interpretability of deep learning models.

It is the goal of this research effort to utilize deep learning to classify galaxy morphologies, and subsequently apply XAI to gain insights on the reliability of the deep learning model's predictions. For our efforts, we adopted SqueezeNet (Iandola et al. 2016) as a test network to classify galaxy morphology images. SqueezeNet has been demonstrably successful in achieving competitive or superior results when compared to prevalent deep learning architectures, such as ResNet and DenseNet, in applications such as lung cancer detection (Tsivgoulis et al. 2022) or wafer defect detection (de la Rosa et al. 2023). Additionally to our best knowledge, there are no publications that report the usage of SqueezeNet to classify galactic morphology. In order to optimize our workflow, we simplified the SqueezeNet algorithm by removing some of the fire modules. In addition, we introduce an improved approach to the simplified SqueezeNet by incorporating additional residual methods into the network which we call Improved SqueezeNet (I-SqueezeNet). We examine whether the residual method helps in improving the performance of the recognition of galaxy morphology. On top of this, we introduce vertical concatenation in the fire module of the simplified SqueezeNet model to further improve its performance.

To quantify the performance of the proposed model, we employed an XAI method called the Local Intepretable Model-

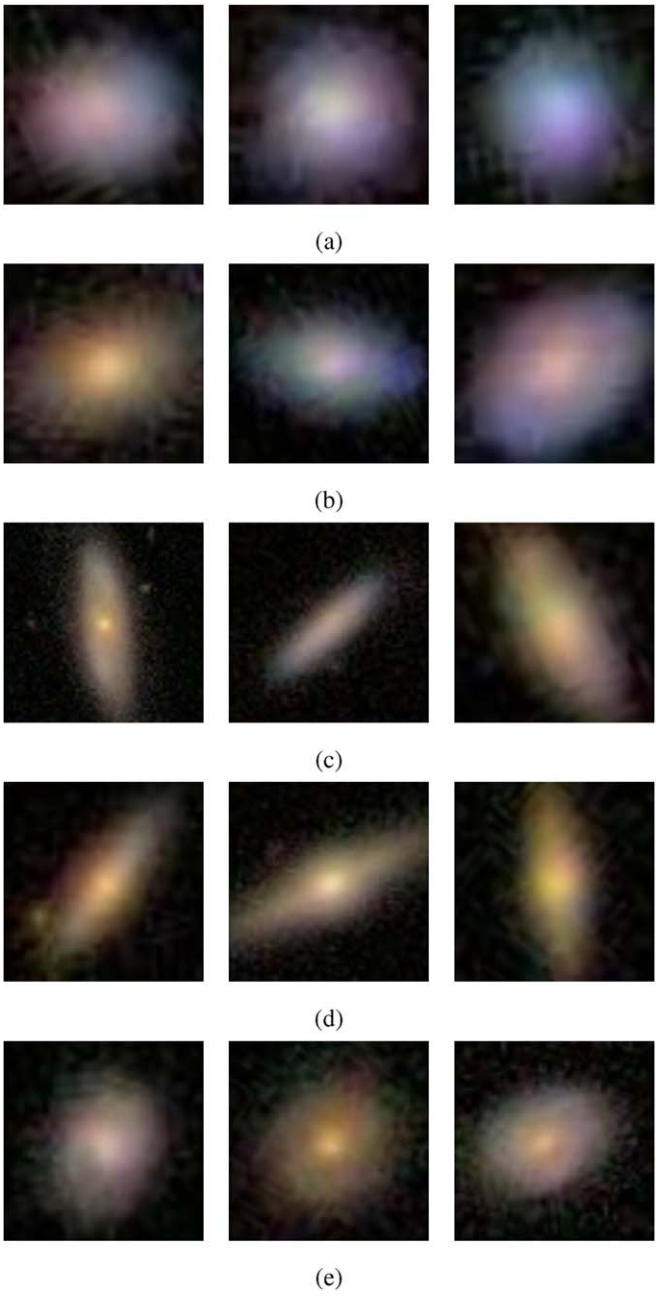

Figure 1. Some samples with similar appearance for (a) completed round smooth samples (b) in-between samples (c) cigar-shaped smooth samples (d) edge-on samples and (e) spiral samples. Some samples of cigar-shaped smooth samples and edge-on samples exhibit visually similar features, while some samples of spiral class and in-between smooth look similar.

Agnostic Explanations (LIME) to interpret and explain the prediction process of our proposed models. The advantage of utilizing LIME is to evaluate the quality of the model's prediction visually based on the trained weights. LIME is a model-agnostic technique applicable to any type of deep learning model. This will address the drawback of the saliency map approaches that are

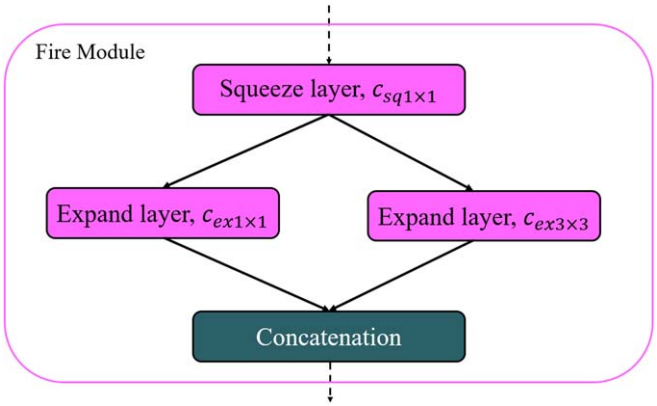

Figure 2. Fire module in SqueezeNet architecture.

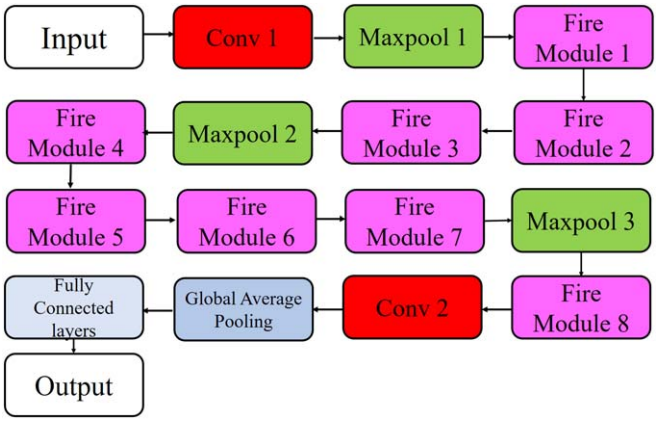

Figure 3. Original SqueezeNet architecture, with two convolutional layers, eight fire modules, and two maxpooling layers.

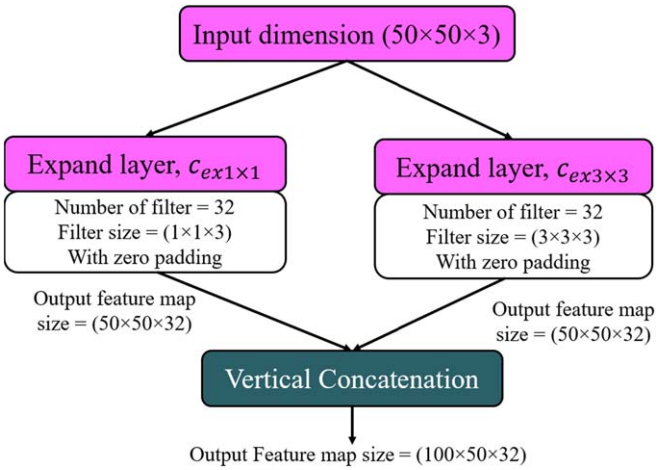

Figure 4. Proposed vertical concatenation.

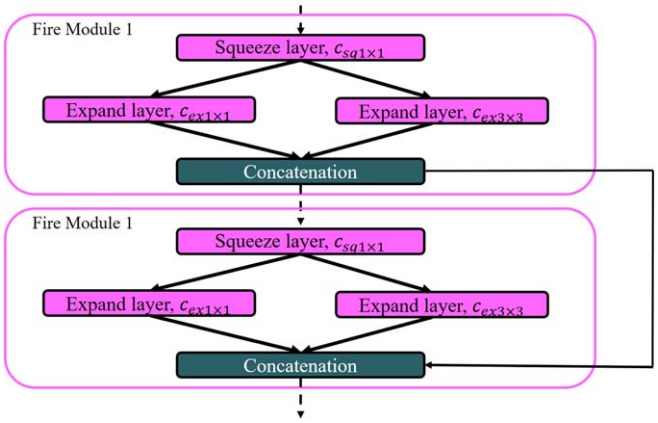

Figure 5. Proposed bypass concatenation.

model-specific. Incidentally we have found no previous work that has applied both the LIME model and SqueezeNet to galaxy morphology classification. With the help of this interpretation, users are able to understand how the machine learning model predicts, and whether the prediction is reliable or questionable. It is also worth noting that LIME is a tool to visualize how the model performs, instead of providing an explanation to improve the prediction via any feedback loop. Our contributions are threefold:

1. Propose a simplified SqueezeNet, and a simplified SqueezeNet with unique residual connection and vertical concatenation for galaxy morphology classification. The well-trained models will be applied to testing dataset to examine their robustness in recognizing galaxy objects.
2. Interpret the simplified SqueezeNet model and its improvements through the use of LIME.
3. Apply LIME to perform quality assessment of the predictions made by various machine learning models.

With the help of our proposed model and LIME explanations, users can leverage this automated machine learning system to classify any star or galaxy object image with interpretability. Instead of blindly trusting the results, users can understand the reasoning behind the predictions. Furthermore, human experts no longer need to examine each image solely based on visual inspection. The remainder of this paper is structured as follows. Section 2 details the dataset used in our study. Section 3 describes the methodology employed for the proposed system. Section 4 presents the experimental results obtained and the major insights we gathered from the result. Finally, Section 5 concludes the study and outlines potential future directions.

2. Data

The Galaxy Zoo Project (Lintott et al. 2008) consists of a crowd-sourced collection of numerous galaxy images. We adopted readily organized 28,793 Galaxy Zoo cropped images restructured by Zhang et al. (2022) with five main classes

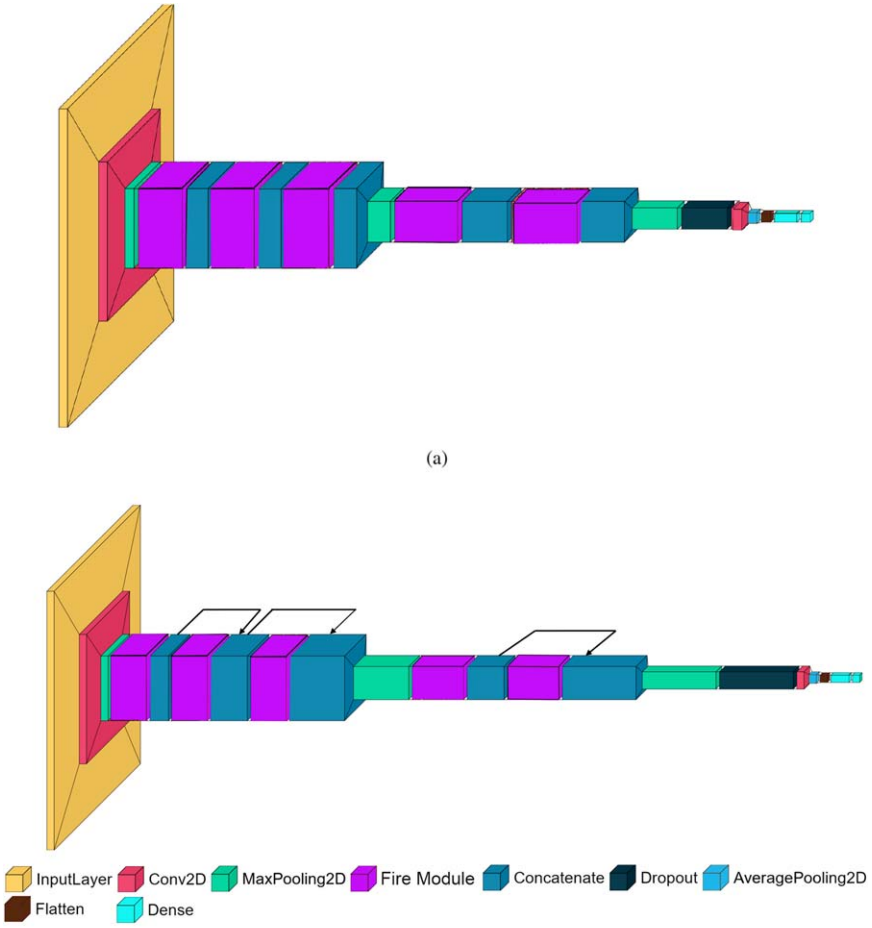

Figure 6. Architecture with channel concatenation for (a) simplified SqueezeNet Architecture, and (b) I-SqueezeNet Architecture.

categorized using these 37 parameters. These five classes include: completely round smooth, in-between smooth, cigar-shaped smooth, edge-on, and spiral. In our study, these classes are designated from class 0 to class 4. These five classes are established based on the filtering method proposed by Zhu et al. (2019), where the annotation of every image is based on the threshold discrimination criteria. The number of samples for every class and respective threshold discrimination criteria are described in Table 1. Some examples of galaxy morphology images are illustrated in Figure 1. It is also worth highlighting that some samples of in-between smooth samples may appear similar to completely round smooth samples or spiral samples, and some samples of cigar-shaped smooth and edge-on may also exhibit similar appearance, which represent a challenge for deep learning models to distinguish between them. Some recent works (Zhu et al. 2019; Urechiatu & Frincu 2024) demonstrated the wrong prediction among these visually similar samples in their confusion matrix.

The dataset was divided, allocating 20% for testing and the remainder for training. Within the training set, 80% was dedicated to actual training, while the remaining 20% served for validation purposes. The same splitting method will be applied across 10 attempts, with the images being randomly split each time—resulting in different training, validation, and test sets for every attempt. All images were resized to 180×180, and a batch size of 32 was employed. We resized all the samples into this dimension for the purpose of enhancing the visibility of important features that were less distinguishable in smaller image sizes. This resolution was selected after several empirical tests, which demonstrated that it provided the best balance between feature visibility, model performance, and also our hardware constraint. Apart from the aforementioned parameters, we applied augmentation to the training dataset to avoid any potential overfitting. The image data augmentation included shearing transformation, rotation, horizontal and vertical shifts, zooming, vertical flip, and

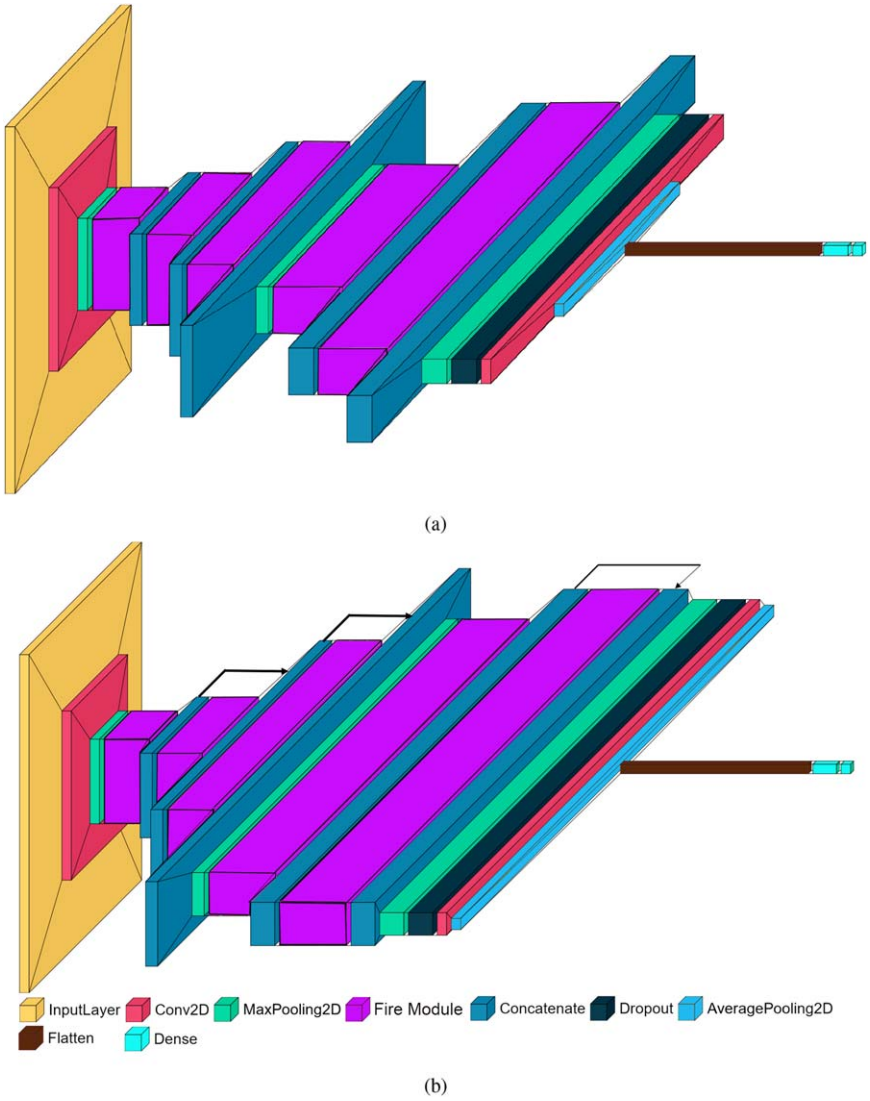

Figure 7. Architecture with vertical concatenation of (a) simplified SqueezeNet Architecture, and (b) I-SqueezeNet Architecture.

horizontal flip. It is worth noting that the augmentation is performed using the TensorFlow Keras function, which is done on-the-fly during training. With this method, the image data are randomly augmented in every epoch when training the model, and the augmented data are not stored on the hard disk, thus not increasing the dataset size.

3. Proposed Method

In this study, our proposed approach involves an improved version of simplified SqueezeNet, namely I-SqueezeNet, followed by implementation of XAI using the LIME model. The I-SqueezeNet serves two vital roles: (1) performing feature extraction through our proposed architecture and (2) performing classification using fully connected (FC) layers, which are based on ANNs. It is important to highlight that the adopted XAI method functions as a standalone post-processing technique applied after the training process. This approach can be applied to any trained CNN model with a single network architecture. Hence, this section explicitly points out the foundational architecture of the simplified SqueezeNet and I-SqueezeNet.

A CNN conventionally consists of two vital layers, the convolutional layer and the maxpooling layer. The convolutional layer performs image filtering to obtain important features that can represent the input images, while the maxpooling layer is utilized to downsample the size of the input and retain the important attributes. Hence, the CNN architecture contains multiple convolutional layers and maxpooling layers for the extraction of various features.

Table 2
Parameters for Simplified SqueezeNet and I-SqueezeNet Architectures

Layer Name	Operation	# of Filters or Nodes	Filter Size/ Pool Size	Activation Function	Stride	Padding
Input	Input	⋯	⋯	⋯	⋯	⋯
Conv1	Convolution	128	7 × 7	Relu	2 × 2	same
Maxpool1	Max-pooling	⋯	3 × 3	⋯	2 × 2	same
Fire_1	Convolution[a]	256	1 × 1	Relu	2 × 2	same
	Convolution[b]	256	1 × 1	Relu	2 × 2	same
	Convolution[b]	256	5 × 5	Relu	2 × 2	same
Merge1	Concatenate layers in Fire_1	⋯	⋯	⋯	⋯	⋯
Fire_2	Convolution[a]	256	1 × 1	Relu	2 × 2	same
	Convolution[b]	256	1 × 1	Relu	2 × 2	same
	Convolution[b]	256	5 × 5	Relu	2 × 2	same
Merge2	Concatenate layers in Fire_2[c] Concatenate Fire_1 and Fire_2[d]	⋯	⋯	⋯	⋯	⋯
Fire_3	Convolution[a]	256	1 × 1	Relu	2 × 2	same
	Convolution[b]	256	1 × 1	Relu	2 × 2	same
	Convolution[b]	256	3 × 3	Relu	2 × 2	same
Merge3	Concatenate layers in Fire_3[c] Concatenate Fire_1, Fire_2 and Fire_3[d]	⋯	⋯	⋯	⋯	⋯
Maxpool2	Max-pooling	⋯	3 × 3	⋯	2 × 2	same
Fire_4	Convolution[a]	128	1 × 1	Relu	2 × 2	same
	Convolution[b]	512	1 × 1	Relu	2 × 2	same
	Convolution[b]	512	3 × 3	Relu	2 × 2	same
Merge4	Concatenate layers in Fire_4	⋯	⋯	⋯	⋯	⋯
Fire_5	Convolution[a]	128	1 × 1	Relu	2 × 2	same
	Convolution[b]	512	1 × 1	Relu	2 × 2	same
	Convolution[b]	512	3 × 3	Relu	2 × 2	same
Merge5	Concatenate layers in Fire_4[c] Concatenate Fire_4 and Fire_5[d]	⋯	⋯	⋯	⋯	⋯
Maxpool3	Max-pooling	⋯	3 × 3	⋯	2 × 2	same
Dropout	Dropout (0.2)	⋯	⋯	⋯	⋯	⋯
Conv2	Convolution	5	1 × 1	⋯	1 × 1	⋯
Average2D	Average Pooling	⋯	2 × 2	⋯	⋯	⋯
Flatten	Flattening	⋯	⋯	⋯	⋯	⋯
Dense1	Dense	500	⋯	Relu	⋯	⋯
Output	Dense	5 (# of classes)	⋯	Softmax	⋯	⋯

Notes.
[a] indicate squeeze layer(s) in fire module.
[b] indicate expand layer(s) in fire module.
[c] is for simplified SqueezeNet.
[d] is for I-SqueezeNet.

Subsequently, these extracted features are flattened into 1D and forwarded to an ANN, commonly known as FC layers, for the purpose of classification. In general, the training process for CNN is to obtain optimal trainable values for filters, weights, and bias within the FC layers to ensure satisfactory performance.

SqueezeNet is one of the well-known networks in CNN, known for its novel fire module. The key contribution of the original SqueezeNet (Iandola et al. 2016) is to incorporate the fire module into the CNN network, for the purpose of reducing trainable parameters while maintaining the efficiency of CNN. The fire module contains multiple convolutional layers. The initial layer is the squeeze convolutional layer, $c_{sq1\times 1}$, where all filters have sizes of 1×1, aiming to reduce parameters. The resulting output is then directed to an expand layer, which includes two parallel-arranged convolutional layers, $c_{ex1\times 1}$, and $c_{ex3\times 3}$, with filters of sizes 1×1 and 3×3, respectively. The layer $c_{ex1\times 1}$ is used to maintain the spatial information and increase the depth of the resulting output from $c_{sq1\times 1}$, while $c_{ex3\times 3}$ is applied to obtain more feature representation using different trainable filters. The output of $c_{sq1\times 1}$ and $c_{ex3\times 3}$ will be concatenated together as an output of the fire

Table 3
Testing Accuracy Using Different Optimizers

Optimizer	Channel Concatenation		Vertical Concatenation	
	Simplified SqueezeNet	I-SqueezeNet	Simplified SqueezeNet	I-SqueezeNet
Adam	29.55%	29.22%	29.57%	30.04%
RMSProp	27.61%	28.18%	28.25%	27.56%
SGD	91.93%	90.42%	**92.09**%	91.07%

module. The fire module is illustrated in Figure 2, and the original architecture of the SqueezeNet proposed by Iandola et al. (2016) is depicted in Figure 3.

As illustrated in Figure 3, the original SqueezeNet architecture consists of two convolutional layers, denoted as Conv1 and Conv2 at the beginning and end of the feature extractors, respectively, before global average pooling layers. It also consists of a total of eight fire modules. Within the original SqueezeNet architecture, concatenation is achieved by combining the feature maps generated by the two expansion layers along the channel dimension. This process results in a new feature map with a depth equal to the sum of the individual expansion layer output channel dimensions. For example, concatenating the outputs of expand layers with 32 and 64 filters would yield a feature map with a depth of 96 channels. In contrast to the original SqueezeNet, our approach proposes vertical concatenation, where the feature maps from the expansion layers are combined along the height dimension. This results in a feature map with a rectangular shape. However, for successful concatenation, the output dimensions of both expansion layers, including the channel dimension, must be equivalent to ensure compatibility. The example of our vertical concatenation is depicted in Figure 4.

To address our hardware limitations, we propose a simplified SqueezeNet architecture that reduces the fire modules from eight to five, while achieving promising performance. Furthermore, we incorporate the concept of residual learning into the network. Unlike conventional approaches in DenseNets or ResNets that utilize bypass connections from earlier layers, our modification leverages a novel concatenation strategy. This strategy involves appending the output of each fire module's concatenation layer to the subsequent module's concatenation layer. This approach ensures the preservation of low-level features extracted by earlier fire modules, which typically capture more localized details. Consequently, the resulting feature maps contain richer information, enhancing the classifier's ability to understand the input image and predict the class label. Figure 5 illustrates the proposed concatenation scheme. Notably, this bypass methodology is applied to both the channel and height dimensions of the simplified SqueezeNet's concatenation layers.

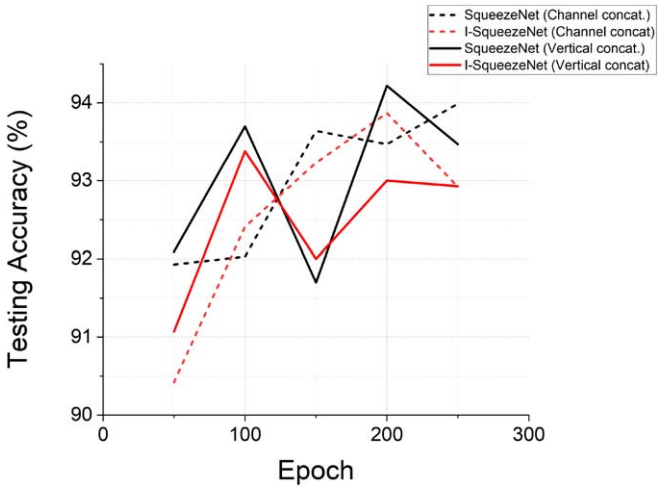

Figure 8. Testing accuracy based on increasing epochs.

The architectures we proposed for our simplified SqueezeNet and I-SqueezeNet for both channel concatenation and vertical concatenation are illustrated in Figures 6 and 7 respectively. All networks contain five fire modules, two convolutional layers, three max-pooling layers, one dropout layer, one average pooling layer, and dense layers. The only difference between I-SqueezeNet and simplified SqueezeNet is the residual concatenation method in the former network. It is important to highlight that both figures explicitly separate the concatenation layer from the fire module to emphasize the depth of the feature maps post-concatenation. Furthermore, architectures employing vertical axis concatenation exhibit an extended dimension along one axis, as illustrated in Figure 7. The optimal parameters of our proposed architectures are demonstrated in Table 2 after empirical testing.

4. Results and Discussions

4.1. Experimental Setup

To examine the effectiveness of our proposed modified network, experiments were performed using the aforementioned dataset, and the performance was quantified using accuracy defined as: (1)

$$\text{acc} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}, \tag{1}$$

where TP, TN, FP, and FN represent the numbers of true positives, true negatives, false positives, and false negatives respectively. Apart from this, we also demonstrated the confusion matrix to check the true positive rate (TPR) for each class. By using the confusion matrix, we aim to identify the classes where the model provided good predictions. In our work, the experiment was performed using TensorFlow and Keras, installed on a machine with 32 GB of RAM, an AMD

Table 4
The Comparison of Testing Accuracy for 10 Attempts

# of Attempt	1st (%)	2nd (%)	3rd (%)	4th (%)	5th (%)	6th (%)	7th (%)	8th (%)	9th (%)	10th (%)	Avg Acc(%)	SD (± %)
Dieleman's CNN	93.34	91.40	92.01	93.09	93.11	92.12	93.11	92.36	93.12	93.42	92.71	0.68
VGG13	88.41	88.61	88.68	88.19	89.49	85.05	87.78	88.42	88.50	87.65	88.08	1.18
DenseNet121	92.93	94.15	94.10	93.47	94.24	94.03	93.49	93.73	94.62	93.67	93.84	0.48
ResNet50	93.24	93.87	93.28	93.56	94.39	93.45	93.87	93.49	94.50	93.56	93.72	0.44
Resxnet50	93.11	92.22	93.14	93.73	94.06	92.95	93.94	93.97	93.84	94.06	93.50	0.62
Resxnet101	94.44	93.87	94.15	93.94	92.59	93.3	93.84	93.52	93.78	94.06	93.75	0.52
Dilated- Separable CNN	89.27	90.26	90.31	91.80	89.51	91.04	91.75	88.90	87.48	90.88	90.12	1.35
Urechiatu's CNN	93.92	93.84	92.85	92.97	94.08	94.53	94.35	93.45	94.28	93.71	93.80	0.57
Simplified SqueezeNet (Channel)	93.74	94.04	93.63	93.35	93.82	93.11	93.38	92.39	93.80	93.25	93.45	0.47
I-SqueezeNet (Channel)	93.38	93.77	93.00	94.37	93.73	94.22	92.95	93.33	93.59	93.56	93.59	0.46
Simplified SqueezeNet (Vertical)	94.04	94.55	93.77	93.82	94.18	93.21	93.56	94.29	93.47	93.80	93.87	**0.40**
I-SqueezeNet (Vertical)	93.91	94.42	93.59	93.97	93.18	94.73	93.89	94.58	94.23	94.32	**94.08**	0.47

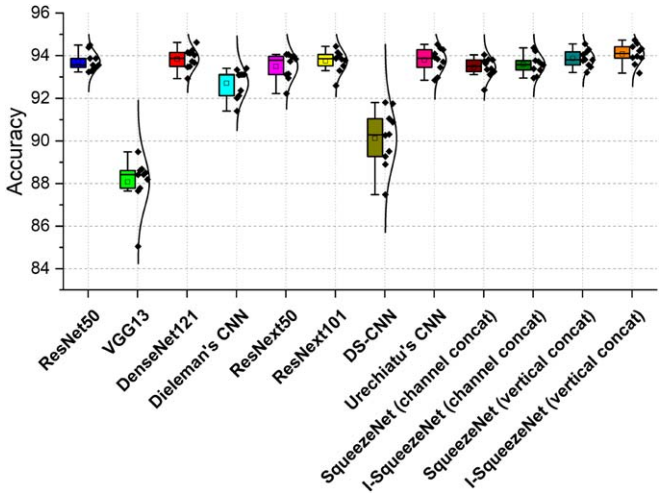

Figure 9. Boxplot of 10 attempts using different methods.

Ryzen 9 5950X 16-Core Processor, and Geforce RTX3090 as our main graphics processing unit (GPU). We first tested the simplified SqueezeNet model and I-SqueezeNet model using different epochs. Additionally, different optimizers were also used to test the model, featuring a learning rate of 0.01. All the processes were tried once to identify optimal setting.

Apart from that, our proposed methods were compared with several recent methods highlighted in 1, such as DenseNet, CNN proposed by Dieleman, VGG, ResNet , and some other recent proposed models. For the training parameters, all models underwent 300 epochs, except for VGG, which was trained for only 150 epochs due to a substantial decline in training accuracy at that point. At the same time, we opted not to compare our work with the Siamese Network (Zhang et al. 2022) because our approach involves a single model rather than executing K-way few-shot learning classification. Each network was trained and tested 10 times, and the average accuracy (Avg acc) and standard deviation (SD) were taken as the final results.

Although the performance of deep learning models has been evaluated, understanding their internal decision-making processes is equally important. To achieve this, we present an analysis of the feature maps extracted from the trained models. In addition, we employ the LIME technique to visualize the interpretability of these models. A detailed discussion of these findings is presented in the subsequent section.

4.2. Hyperparameter Testing

We initially trained both the simplified SqueezeNet and I-SqueezeNet models using various optimizers, including Adam, RMSProp, and Stochastic Gradient Descent (SGD). We subsequently compared the simplified SqueezeNet with our modified version that consists of vertical concatenation. These optimizers are gradient-based approaches, and they are commonly used in neural networks to update weights and biases according to the cost function throughout iterations. However, each of these three optimizers employs its unique method to adapt the learning rate throughout the computation process. The number of epochs was set at 50 and the initial learning rate was set to 0.01. The testing accuracies are shown in Table 3, and the highest value is in bold.

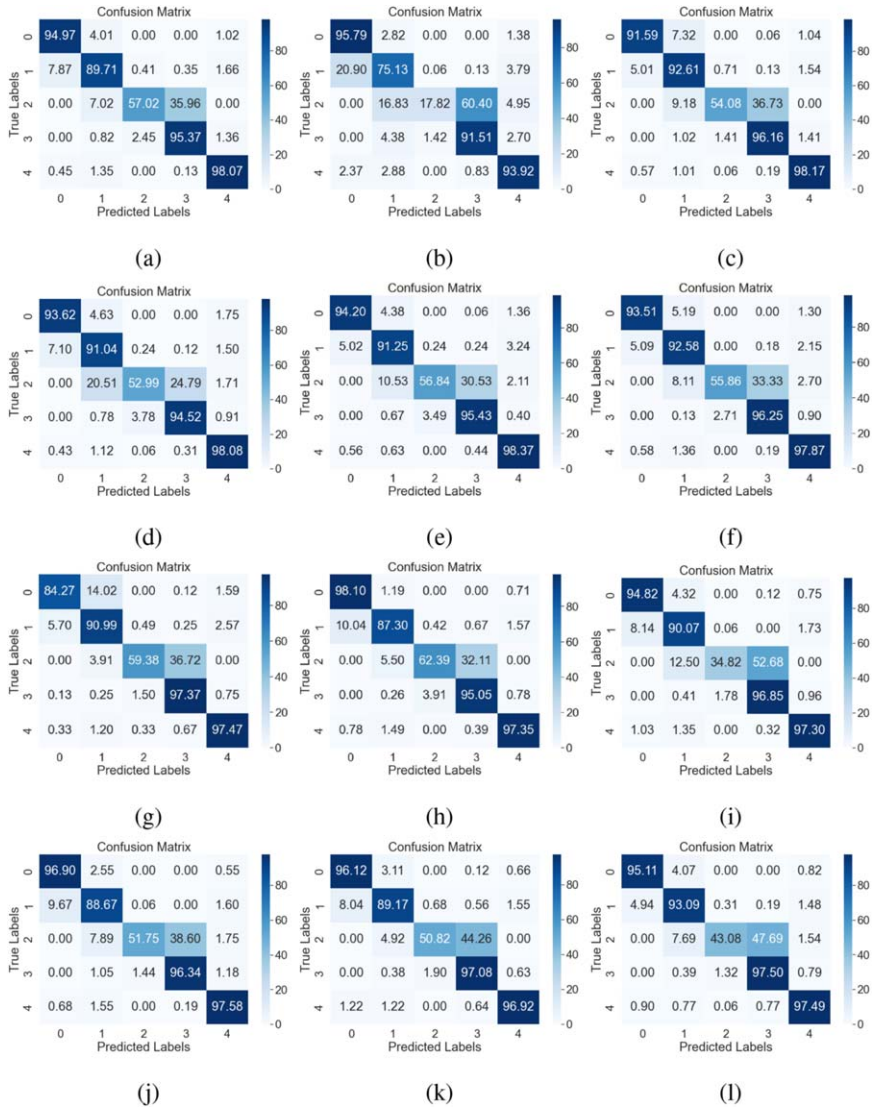

Figure 10. Confusion matrix for (a) ResNet50, (b) VGG13, (c) DenseNet, (d) Dieleman's CNN, (e) ResNext50, (f) ResNext101, (g) DS-CNN, (h) Urechiatu's CNN, (i) simplified SqueezeNet with channel concatenation, (j) simplified SqueezeNet with vertical concatenation, (k) I-SqueezeNet with channel concatenation, and (l) I-SqueezeNet with vertical concatenation.

Table 3 demonstrates that SGD achieves superior convergence compared to Adam and RMSprop optimizers. Adam and RMSprop might be susceptible to local minima, hindering training progress. Notably, SGD yields the highest testing accuracy for both channel and vertical concatenations. In vertical concatenation, I-SqueezeNet achieves a minimum accuracy of 91%, while simplified SqueezeNet reaches an impressive 92.09% accuracy. Channel concatenation exhibits slightly lower accuracies, with simplified SqueezeNet and I-SqueezeNet achieving 91.93% and 90.42% respectively. This demonstrates that vertical concatenation may surpass channel concatenation in extracting important features during the training process.

Apart from aforementioned results, we further examined the effectiveness of both simplified SqueezeNet and I-SqueezeNet (with both concatenation) alongside the SGD optimizer across varying epochs, ranging from 50 to 250 with increments of 50. The evaluation focused on testing accuracy, as depicted in Figure 8. The results indicate a noticeable upward trend in accuracy for all models over the epochs considered. Moreover, both models demonstrated comparable results, with simplified SqueezeNet surpassing I-SqueezeNet in most of the epochs. Drawing from the empirical testing conducted in this section, we extended our comparisons by evaluating our methods against other approaches across 300 epochs. All methods

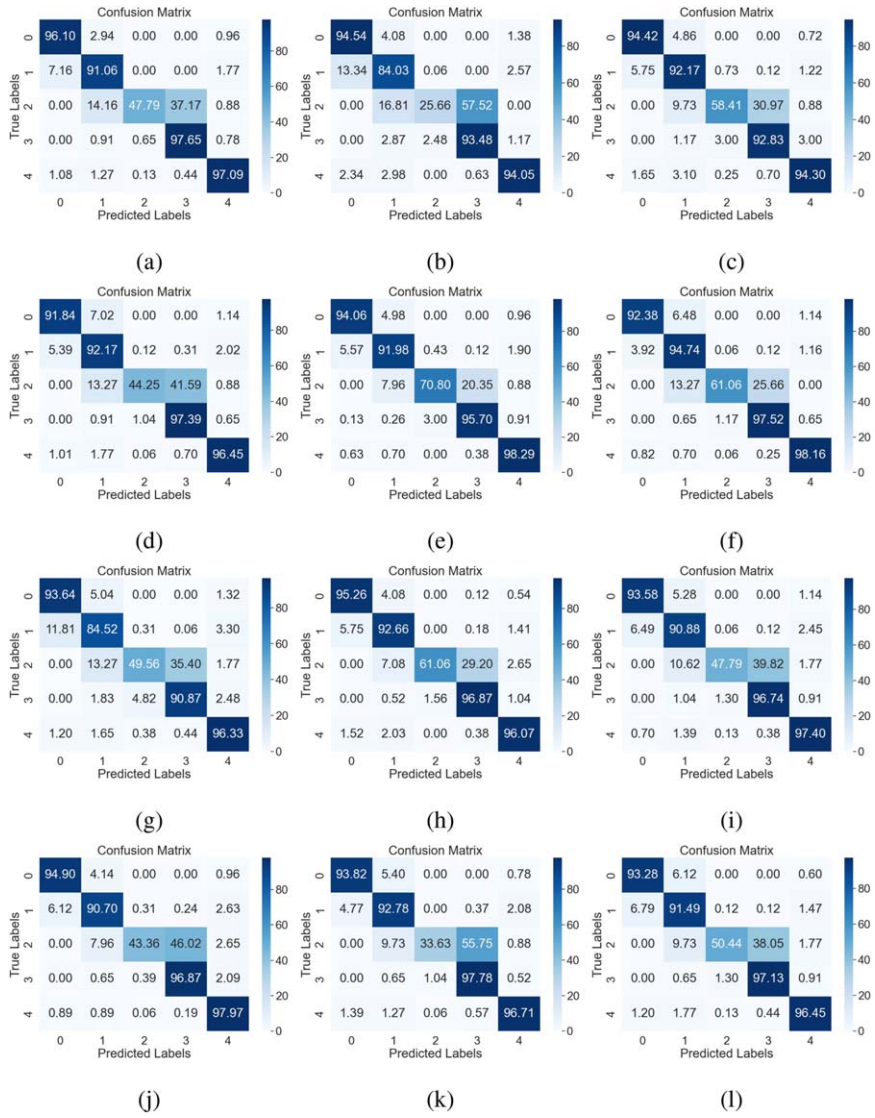

Figure 11. Confusion matrix under same training/testing dataset for (a) ResNet50, (b) VGG13, (c) DenseNet, (d) Dieleman's CNN, (e) ResNext50, (f) ResNext101, (g) DS-CNN, (h) Urechiatu's CNN, (i) simplified SqueezeNet with channel concatenation, (j) simplified SqueezeNet with vertical concatenation, (k) I-SqueezeNet with channel concatenation, and (l) I-SqueezeNet with vertical concatenation.

employed SGD optimizer for weight and bias update. The comparison is thoroughly discussed in the subsequent section.

4.3. Comparison with Other Methods

As mentioned earlier, simplified SqueezeNet and I-SqueezeNet were compared to other recent methods. The performances are depicted in Table 4, with the bolded highest accuracy and lowest SD. The results demonstrated the effectiveness of simplified SqueezeNet and I-SqueezeNet with both concatenations, where average accuracies obtained from these four models surpassed the other methods. According to Table 4, the performance of simplified SqueezeNet architectures with channel concatenation was comparable to DenseNet121, ResNext101 and Urechiatu's CNN with an average accuracy of 93.45%. Similar results were obtained using the alternative concatenation method (93.87%). Additionally, it was observed that the simplified SqueezeNet with vertical concatenation produced more consistent results throughout the entire 10 attempts, with a standard deviation of 0.40%, which is relatively lower than those of other models. Meanwhile, I-SqueezeNet with vertical feature map concatenation reached the highest average accuracy among the models, achieving an average accuracy of 94.08%. It is also worth emphasizing that

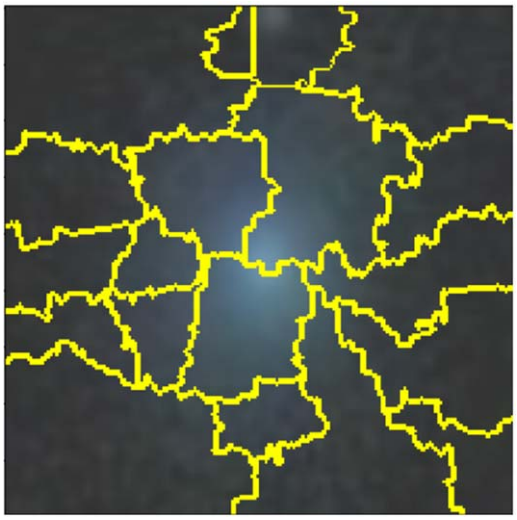

Figure 12. Galaxy morphology image with superpixels.

I-SqueezeNet with vertical concatenation achieved its highest accuracy during the 6th attempt, reaching 94.73%. Notably, the vertical concatenation approach yielded superior results compared to channel concatenation. However, the performance difference between simplified SqueezeNet and I-SqueezeNet is not significant, where the difference of the average accuracy was only about 0.14% and 0.21% for channel concatenation and vertical concatenation respectively.

To ease the visualization of the results, we have generated a boxplot with normal distribution chart as illustrated in Figure 9. From this figure, it is evident that DenseNet121, ResNext101, simplified SqueezeNet, and I-SqueezeNet demonstrated comparable performance, which showed the positive impact of residual connection. Our analysis also revealed a positive correlation between the performance with mild improvement observed when transitioning from simplified SqueezeNet to I-SqueezeNet with the modifications implemented to the concatenation method. It is also evident that I-SqueezeNet with vertical feature map concatenation yielded the best performance. In contrast, DS-CNN and VGG13 lacked residual connections, leading to inferior performance. Moreover, the normal distribution curves of these two models are relatively broader compared to those with residual connections due to higher standard deviations, suggesting instability and also lower reproducibility for these models.

We were eager to investigate the prediction performance for specific classes of galaxy objects for each model. The details are depicted by the confusion matrix for each method in Figure 10. It is worth noting that these confusion matrices are generated from the trained model obtained from the last attempts (10th attempt) listed in Table 4. Since the standard deviation is small across all the attempts, these also indicate that the confusion matrices from different attempts exhibit only minor variations. Since they are largely similar, presenting just one is sufficient to represent the model's performance. The numbers shown along the diagonal of the confusion matrix indicate the TPR for each class.

An analysis of Figure 10 reveals a consistent decrease in performance across all models for samples classified as class 2 (cigar-shaped smooth galaxies). This is likely due to a class imbalance, with class 2 containing a substantially lower number of samples compared to other classes. The TPR for class 2 ranged from 40% to 63% for all models, with the exception of the VGG13 model, which achieved a TPR of only 17.82%. Furthermore, it is noteworthy that all models exhibited difficulty in differentiating class 2 from class 3. These classes share a high degree of similarity in visual appearance or characteristic features, potentially leading to misclassifications. Meanwhile, proposed simplified SqueezeNet architectures achieved prominent performance in classifying class 0 samples. The simplified SqueezeNet employing vertical concatenation attained a superior TPR of 96.90%. Nonetheless, Urechiatu's CNN achieved at least 98% for recognizing class 0 samples. Additionally, the I-SqueezeNet architecture with vertical concatenation demonstrated a distinct advantage in classifying class 1 objects, as reflected in its TPR of 93.03%. Our proposed modifications, which incorporate residual connections and vertical concatenation, demonstrably improved the performance of the simplified SqueezeNet architecture. This is reflected in the significantly higher TPR achieved by our model compared to the simplified SqueezeNet without skipped connection (90.07%). While the proposed simplified SqueezeNet and I-SqueezeNet architectures achieve promising results for galaxy morphology classification, their performance on class 4 falls marginally below that of DenseNet, ResNet, ResNext50, and Dieleman's network. These architectures achieved a TPR of at least 98%. Despite this, the modified SqueezeNet series maintains a respectable and acceptable TPR range of 96.92%–97.58% for classifying class 4 samples. This study demonstrates that our proposed architecture achieves performance on par with, and in some cases exceeding, that of current existing approaches. However, the proposed skip connection exhibited certain weaknesses, particularly when handling samples from class 2, which led to a lower TPR. This may be attributed to the visually similar features shared between class 2 objects and those of other classes. This indicates that there is still room for improvement in the proposed architecture, especially when dealing with objects with similar appearance features. To further validate the results, we conduct one additional experiment so that all networks were evaluated using the same training, validation, and testing datasets to ensure a fair comparison. Additionally, early stopping was implemented to prevent overfitting. The confusion matrices of all the testing results are illustrated in Figure 11

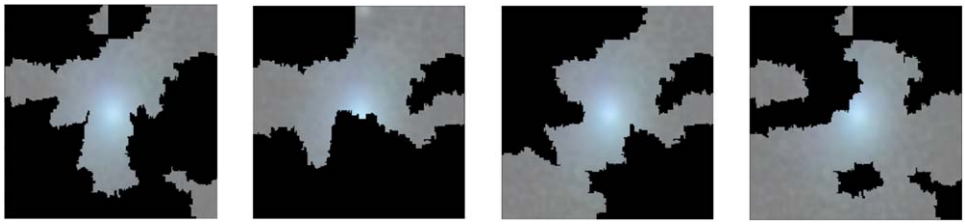

Figure 13. Example of perturbed images.

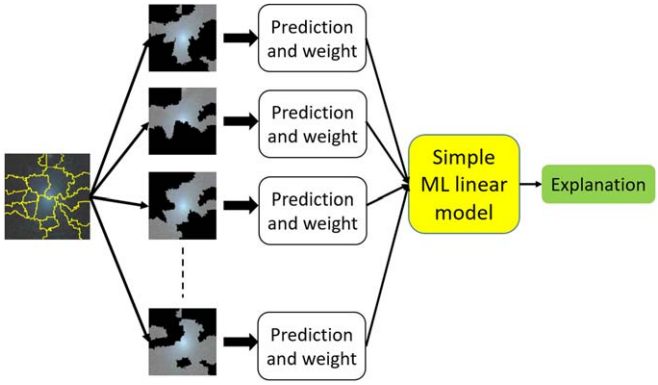

Figure 14. Pipeline of LIME model.

Figure 11 illustrates that all models struggle to distinguish between class 2 and class 3 objects. However, our proposed modified SqueezeNet and its variants successfully identify class 3 objects, achieving a TPR of at least 97%. It is worth noting that the I-SqueezeNet with channel concatenation delivers the highest TPR for class 3 compared to other methods, reaching 97.78%. For class 0, class 1, and class 4, the TPRs of our proposed variants are on par with other methods, indicating consistent performance across classes. We believe the use of early stopping may have limited the models from achieving even higher prediction scores. Further discussion on visually similar objects and model interpretability will be presented in the following section.

4.4. LIME Visualization and Interpretability

To understand the prediction rationale behind our machine learning model, we employed the LIME algorithm for interpretability analysis on five selected test samples. LIME (Ribeiro et al. 2016) is broadly known for understanding how a black box predicts. This approach seeks to estimate the predictions made by a trained model by employing straightforward and interpretable machine learning techniques, such as linear regression. To achieve this, the LIME approach creates explanations by initially generating superpixels are generated from the input images. An example of galaxy morphology images with superpixels is demonstrated in Figure 12. In certain images, perturbation is performed, where the intensities of these superpixels are subsequently modified to zeros randomly, and the trained deep learning model is employed to predict the outcomes of these altered images. Some examples of the perturbed images can be found in Figure 13.

Figure 12 illustrates the segmentation of a galaxy morphology image into visually distinct superpixels, delineated by yellow lines. These superpixels are formed by clustering pixels with similar characteristics. The yellow lines outline these superpixel regions, which serve as the foundation for LIME's perturbation analysis. By isolating these regions using a yellow line, LIME can examine the impact of a specific image segment on the model's predictions, hence showing the interpretability.

Prior to formulating the explanation, weights were initially derived by considering the distance between the predictions made on the original images and those on perturbed images. A higher weight assigned to a perturbed image signifies that the perturbation is closer to the original image. Ultimately, these weights, along with the prediction results and each perturbed image, serve as inputs to train a basic interpretable machine learning model used to explain the deep learning model. In general, LIME aims to discover the local fidelity of interpretable representations which are locally faithful to the classifiers. The general pipeline of the LIME model is shown in Figure 14.

Mathematically, the explanation can be expressed by (2):

$$\xi(x) = \underset{g \in G}{\operatorname{argmin}}\, \mathcal{L}(f, g, \pi_x) + \Omega(g), \tag{2}$$

where ξ represents the explanation, f denotes the prediction probability of our proposed deep learning model for the instance of interest (represented by x), g is the interpretable model, a member of a set of possible explanations G, and $\Omega(g)$ signifies the complexity level of the interpretable model. The proximity measure π_x indicates the locality or neighborhood surrounding x that we can use for explanation. In short, the goal

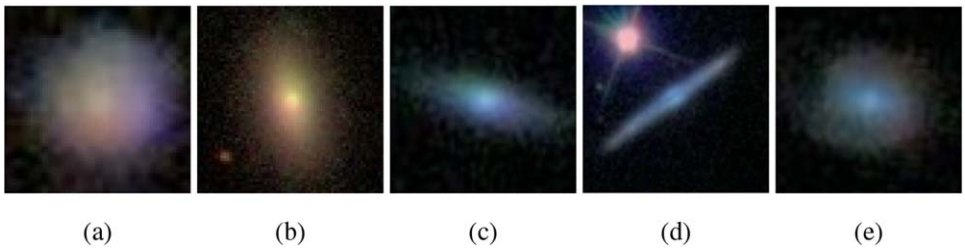

Figure 15. Samples from (a) class 0, (b) class 1, (c) class 2, (d) class 3, and (e) class 4.

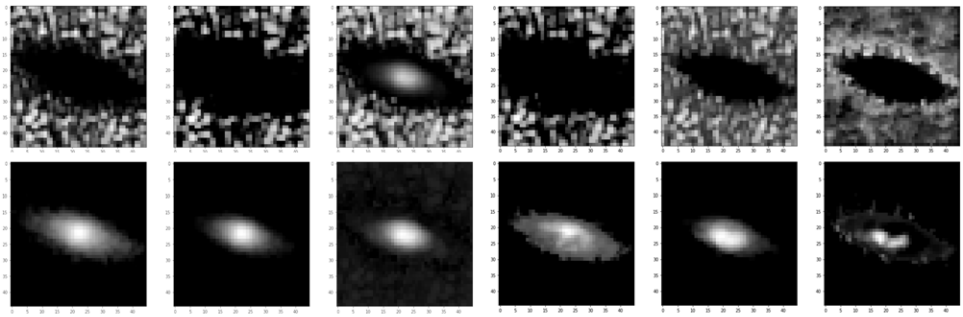

Figure 16. Feature maps from (a) simplified SqueezeNet with channel concatenation, and (b) I-SqueezeNet with channel concatenation.

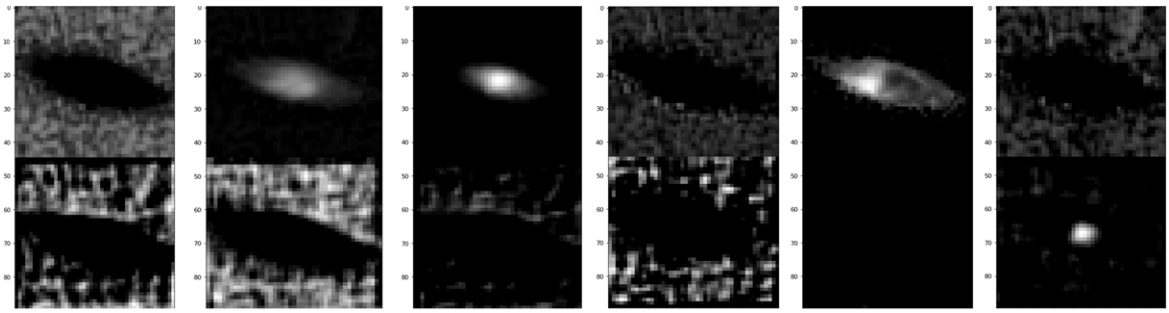

Figure 17. Feature maps from (a) simplified SqueezeNet with vertical concatenation, and (b) I-SqueezeNet with vertical concatenation.

of the LIME model is to minimize the loss $\mathcal{L}$ produced by g while simultaneously keeping the complexity low. In this paper, the ridge regressor is used as the weighted linear model. In the end, the LIME model will produce a mask that points out the region that visually contributes to the prediction.

To visualize the prediction, we strategically selected one sample each from every class, as depicted in Figure 15. To gain insights into the learned features, we initially investigated the filter outputs (feature maps) generated by all the trained simplified SqueezeNet and I-SqueezeNet architectures that we proposed in our work. We selected the six most significant feature maps obtained from the concatenation layer (Merge_1) after the first fire module qualitatively. We demonstrate the feature maps of a class 2 object from Figure 15(c) for simplified SqueezeNet and I-SqueezeNet with channel concatenation in Figure 16.

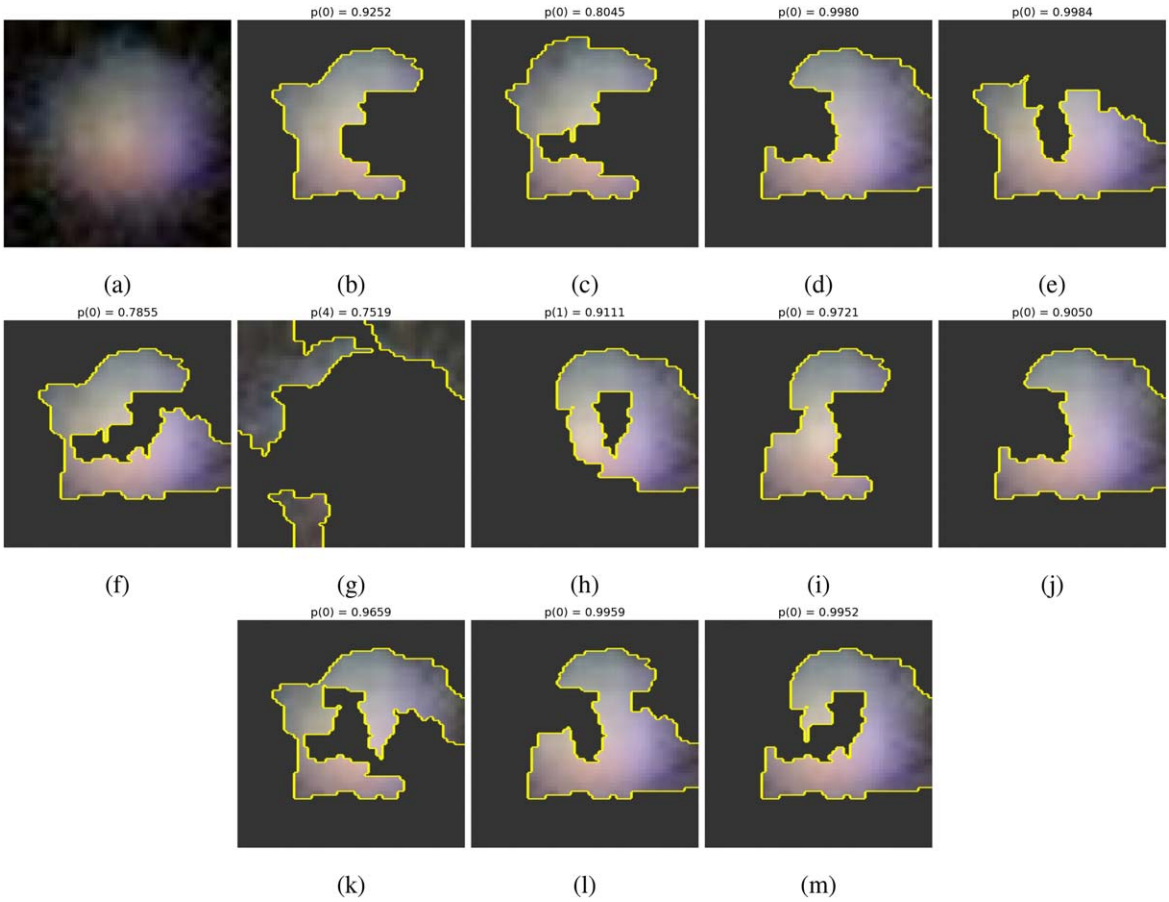

Figure 18. (a) Original image of class 0 object, (b) LIME explanation of Dieleman's CNN, (c) VGG13, (d) DenseNet, (e) ResNet, (f) ResNext50, (g) ResNext101, (h) DS-CNN, (i) Urechiatu's CNN, (j) simplified SqueezeNet (channel), (k) I-SqueezeNet (channel), (l) simplified SqueezeNet (vertical), and (m) I-SqueezeNet (vertical).

As illustrated in Figure 16, the trained model successfully captured the inclined shape of the galaxy object, which may resemble features of a class 3 object, potentially leading to misclassification. The concatenation operation in Merge_1 is channel-wise, preserving the spatial dimensions of the individual feature maps. In addition, some of the feature maps may not carry important information, which may be discarded by the activation function in later layers. The feature maps generated from simplified SqueezeNet and its variants with vertical concatenation are shown in Figure 17.

Based on Figure 17, the output has resulted in a unified rectangular feature map due to spatial concatenation. Moreover, the analysis revealed that certain feature maps integrate information from distinct spatial regions (object and surroundings). This necessitates the training process to develop optimal filters that effectively capture these complementary features. It is noteworthy, however, that the first fire module employs a total of 256 filters across the squeeze and expand convolutional layers. While this investigation focuses on a select few feature maps for qualitative analysis and discussion, the complete 256 set may offer a more comprehensive understanding of the learned features.

While feature map visualization offers valuable insights, analyzing the weights and biases associated with the hidden layers (particularly the dense1 layer) is crucial. This analysis can reveal which elements within the flattened feature vectors contribute most significantly to the network's decision-making process. Given the model's multilayer perceptron architecture and the inherent nonlinearity of its decision-making process, we opted to leverage LIME for interpretability analysis. We demonstrated the LIME explanations of all samples in Figure 15 for all models in Figures 18, 19, 20, 21, and 22. It is also worth noting that the prediction probabilities can be found on the top of all the subimages of all these figures, and they are expressed as $p(x)$, where x represents the class index. This enables a direct comparison of the interpretability between our models and recent methods.

For class 0 samples, most models successfully predict the class with high probabilities, except for ResNeXt101 and DS-CNN, as shown in Figure 18. From the visualization, DenseNet

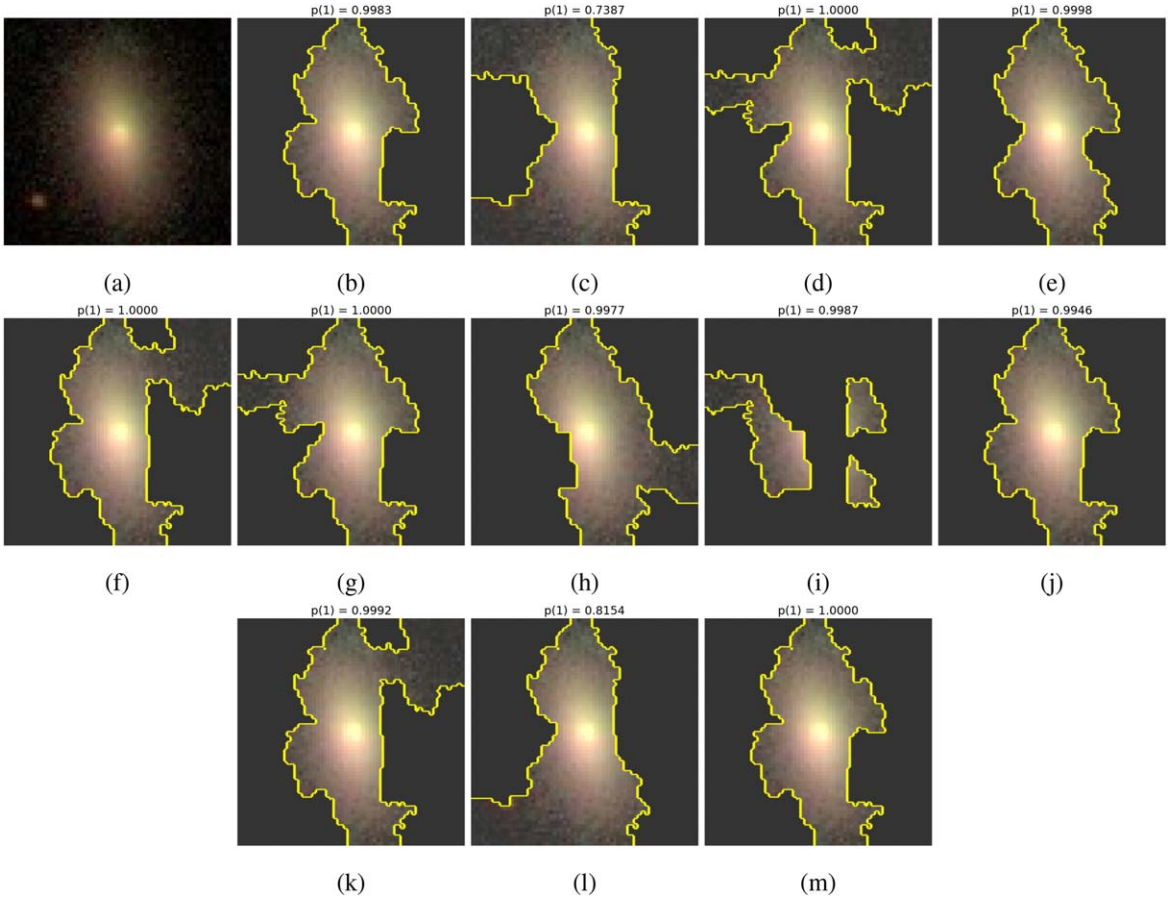

Figure 19. (a) Original image of class 1 object, (b) LIME explanation of Dieleman's CNN, (c) VGG13, (d) DenseNet, (e) ResNet, (f) ResNext50, (g) ResNext101, (h) DS-CNN, (i) Urechiatu's CNN, (j) simplified SqueezeNet (channel), (k) I-SqueezeNet (channel), (l) simplified SqueezeNet (vertical), and (m) I-SqueezeNet (vertical).

and simplified SqueezeNet with channel concatenation provide identical explanations, as their LIME outcomes are exactly the same. Notably, most trained models focus on the middle region of the sample, implying that the middle region of this class contains the key features driving the prediction. A similar observation can be made for the class 1 sample, as shown in Figure 19, where most LIME explanations emphasize the central vertical region as the area containing important features. Interestingly, Dieleman's CNN, the simplified SqueezeNet with channel concatenation, and I-SqueezeNet with vertical concatenation share identical LIME explanations for this sample. Meanwhile, ResNeXt50 and I-SqueezeNet with channel concatenation exhibit similar visualizations. Additionally, all models correctly predicted this class with an accuracy of at least 70%, with Urechiatu's CNN having slightly different LIME explanation. Overall, most of the models are able to highlight the key region of class 0 and class 1 samples for interpretation.

Consistent with the confusion matrices presented in Figure 10, class 2 objects exhibited a misclassification pattern, where they were always predominantly predicted as class 3. Class 2 objects are susceptible to misclassification as class 3 due to their visual similarity. To investigate this further, all models were employed alongside LIME to predict and explain the sample depicted in Figure 15(c). As illustrated in Figure 20, with the exception of VGG13 (41.32% probability) and DS-CNN (53.17% probability), most models classified the sample wrongly as either class 1, class 3 or class 4. The LIME explanations (Figure 20) reveal that DenseNet, ResNet, DS-CNN and simplified SqueezeNet with vertical concatenation relied on the inclined region for their predictions. Moreover, I-SqueezeNet with vertical concatenation completely excluded the galaxy object region for prediction. Interestingly, while the feature maps extracted for classification (Figures 16 and 17) captured the galaxy object regions, LIME suggests that these features were only partially utilized by the neural network. It is important to acknowledge that the superpixel segmentation employed by the LIME model exhibits inherent randomness. This means that the specific superpixels generated may differ across model executions. In addition to the aforementioned

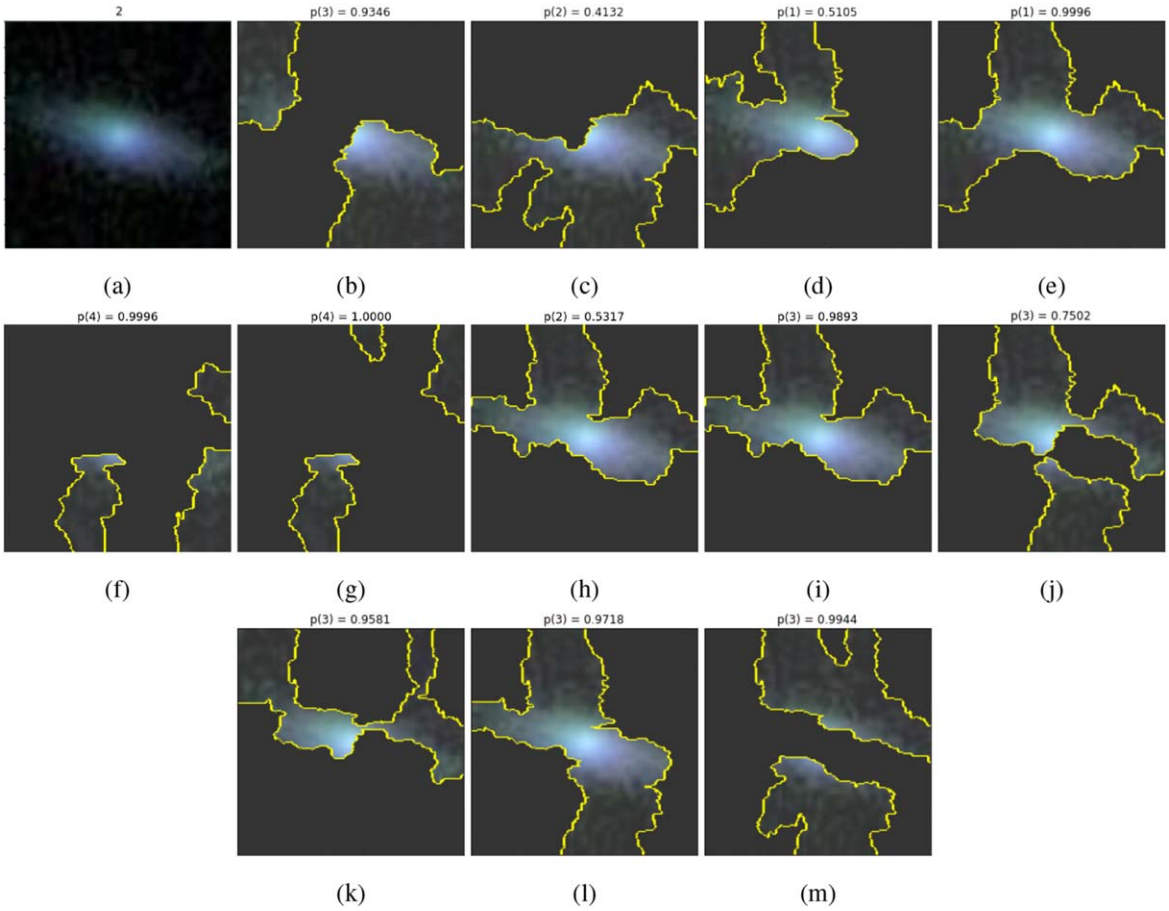

Figure 20. (a) Original image of class 2 object, (b) LIME explanation of Dieleman's CNN, (c) VGG13, (d) DenseNet, (e) ResNet, (f) ResNext50, (g) ResNext101, (h) DS-CNN, (i) Urechiatu's CNN, (j) simplified SqueezeNet (channel), (k) I-SqueezeNet (channel), (l) simplified SqueezeNet (vertical), and (m) I-SqueezeNet (vertical).

visualization, we also illustrate the explanation of class 3 and class 4 objects in the following descriptions.

Most models achieved high prediction accuracy (above 80%) for the correct prediction of the sample from class 3, as shown in Figure 21. The explanations associated with these predictions (refer to Figure 21(c) and (g)) reveal that, with the exception of the VGG13 model and ResNext101, all models identified almost the entire star object as a critical feature for classification. In contrast, the VGG13 and ResNext101 model assigned importance only to a limited central region of the object for predicting the galaxy object. In addition to this, the prediction probability of VGG13 was only 81.32%. Interestingly, Dieleman's CNN and I-SqueezeNet architectures, both employing concatenation, demonstrably achieved 100% accuracy in predicting the sample. We therefore can conclude that the inclusion of our proposed skipped connection further strengthens the model's performance. It is also interesting to point out that ResNext101 predicted the class of this sample as a class 4 object with 100% score, which also hinted at the possibility of wrong prediction using advanced networks.

As shown in Figure 22, Dieleman's CNN and simplified SqueezeNet configured with vertical concatenation misclassified the class 4 sample as class 0. Conversely, all other models achieved accurate classifications. Notably, VGG13, simplified SqueezeNet with channel concatenation, and both I-SqueezeNet variants (vertical and channel concatenation) assigned high importance to the entire central region of the sample for their predictions. While DenseNet, both ResNexts, and ResNet correctly classified the sample, the LIME explanations indicate that these models did not rely on the central region for classification, particularly DenseNet. Hence, these results raise concerns about a potential disconnect between model performance and trustworthiness. While the models achieved accurate classifications, the LIME

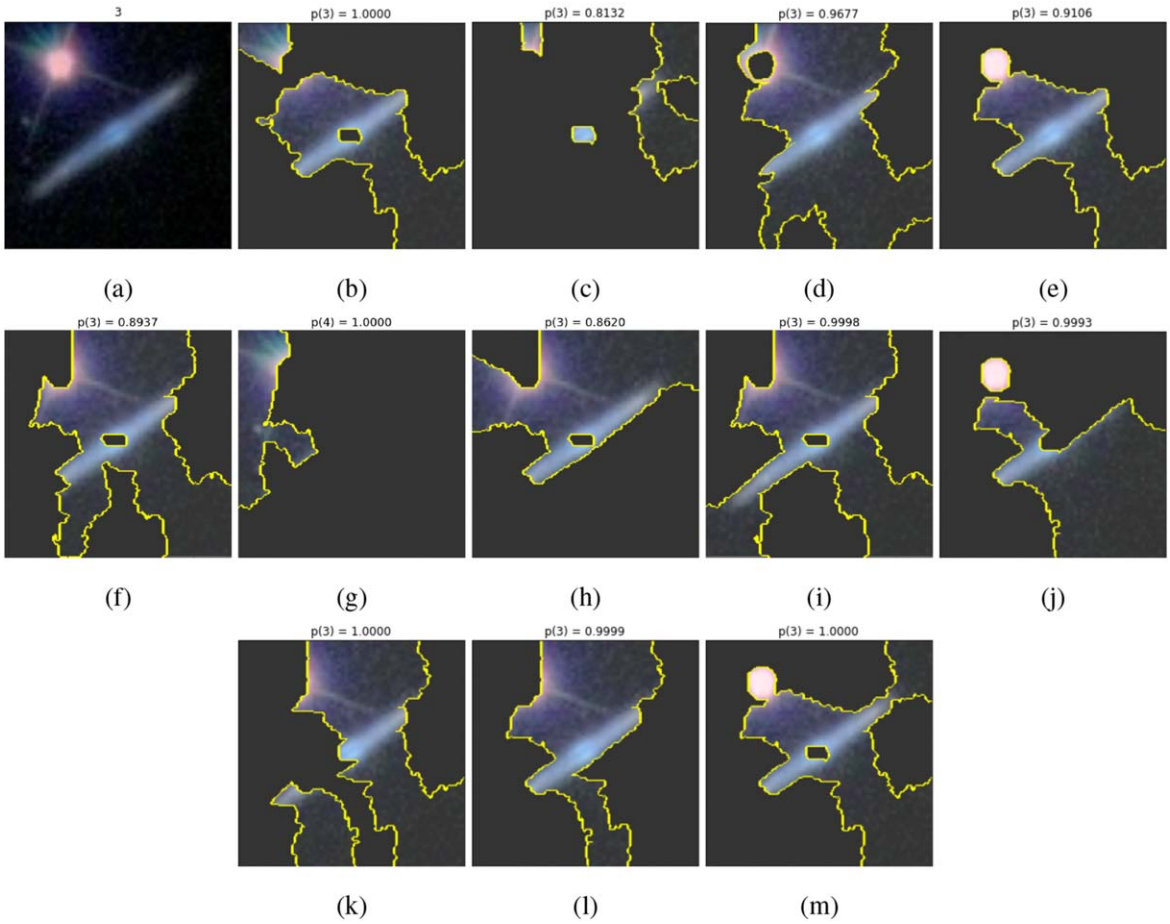

Figure 21. (a) Original image of class 3 object, (b) LIME explanation of Dieleman's CNN, (c) VGG13, (d) DenseNet, (e) ResNet, (f) ResNext50, (g) ResNext101, (h) DS-CNN, (i) Urechiatu's CNN, (j) simplified SqueezeNet (channel), (k) I-SqueezeNet (channel), (l) simplified SqueezeNet (vertical), and (m) I-SqueezeNet (vertical).

explanations suggest they might not be relying on the most relevant features for their judgements.

The findings from this study, informed by LIME explanations, highlight the importance of interpretability even for high-performing models. While some models achieved exceptional results, the LIME analysis revealed potential shortcomings in how they arrived at their predictions. This underscores the necessity of interpretability for ensuring the validity and trustworthiness of model predictions. Meanwhile, one shortcoming of the LIME model is its inability to offer quantitative metrics for evaluating its explanation effectiveness. While it provides visual explanations, these lack numerical measures to objectively assess the quality of the explanation.

5. Conclusions

Traditional methods for classifying galaxy morphology rely on years of expert knowledge, introducing potential human bias and error. While both parametric and non-parametric algorithms have been developed, computational limitations remain a significant challenge. Recent advancements in computer vision and deep learning offer promising avenues to address this issue. Several deep learning techniques have been proposed to automate the classification process; however, the transparency of these models within the context of galaxy morphology classification remains an under-investigated area. As a result, we employed the LIME model to gain qualitative insights into various deep learning models' predictions, including our proposed approach that builds upon the simplified SqueezeNet architecture and incorporates a novel enhancement designed to improve galactic morphology recognition. The performance of modified SqueezeNet and its variants, as well as the other recent models were comprehensively investigated. Our studies recorded the best average accuracy of 94.08%, accompanied by a low standard deviation of 0.47% achieved by I-SqueezeNet with vertical concatenation. Moreover, among the investigated configurations, I-SqueezeNet with vertical setting demonstrated the best

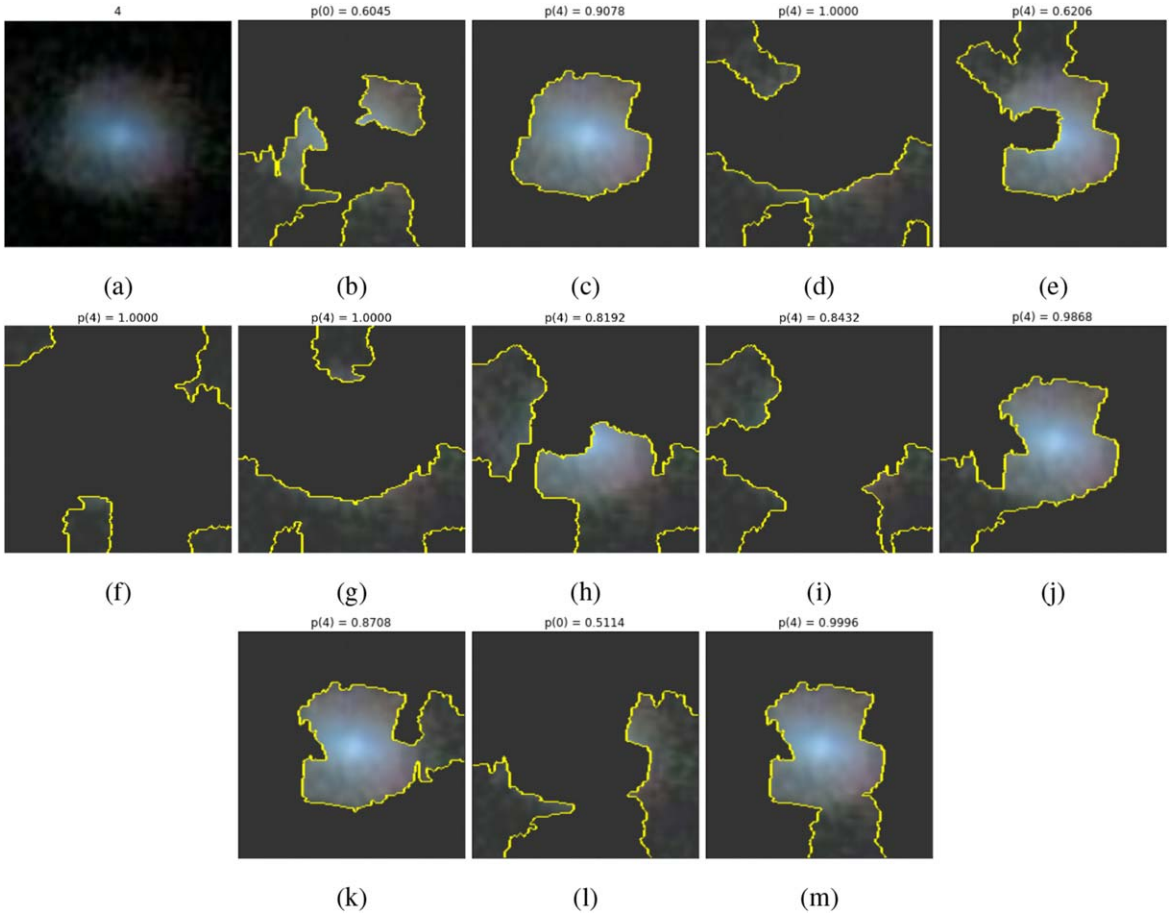

Figure 22. (a) original image of class 4 object, (b) LIME explanation of Dieleman's CNN, (c) VGG13, (d) DenseNet, (e) ResNet, (f) ResNext50, (g) ResNext101, (h) DS-CNN, (i) Urechiatu's CNN, (j) simplified SqueezeNet (channel), (k) I-SqueezeNet (channel), (l) simplified SqueezeNet (vertical), and (m) I-SqueezeNet (vertical).

performance, reaching 94.73% accuracy in an experimental run. On the other hand, I-SqueezeNet with channel concatenation showed its effectiveness in classifying objects from class 1, with TPR of 93.09%. Overall, the LIME model enabled us to effectively identify the image regions most influential for the model's classification. The demonstrated qualitative results suggest that the deep learning model might utilize unexpected image regions beyond the galaxy object itself for classification.

Nevertheless, an imbalanced dataset can deteriorate the performance of the proposed models. It is also worth noting that our proposed skip connection does not help in improving the TPR for a class 2 object, which implies a shortcoming of our proposed method. Looking toward future improvements, the proposed method could benefit from the incorporation of an attention mechanism. This integration would promote a stronger focus on the key regions crucial for galactic morphology recognition. We are cautiously optimistic that the integration of an attention mechanism can help in achieving better TPR for every class, as well as providing necessary transparency. Furthermore, stratified data balancing can be employed to address the imbalanced data concern. Additionally, exploring other model-specific XAI techniques, such as analyzing the gradients of specific model layers, could provide further insights into the significant regions used for classification. Moreover, the proposed classification method can be potentially applied in any other galaxy tasks related to imaging.

Acknowledgments

The CNN architecture in Figures 6 and 7 was illustrated using Visual Keras by Gavrikov (2020).

Data

The data used in this work are based on the dataset provided in Zhang et al. (2022). Code can be requested by contacting the corresponding author.

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.

Funding

No fund was received for conducting this work.

ORCID iDs

Kam Meng Goh https://orcid.org/0000-0003-0378-7390
Kolla Bhanu Prakash https://orcid.org/0000-0002-7955-2777

References

Abraham, R. G., van den Bergh, S., & Nair, P. 2003, ApJ, 588, 218
Baillard, A., Bertin, E., De Lapparent, V., et al. 2011, A&A, 532, A74
Banerji, M., Lahav, O., Lintott, C. J., et al. 2010, MNRAS, 406, 342
Barchi, P., da Costa, F., Sautter, R., et al. 2017, arXiv:1705.06818
Barchi, P., de Carvalho, R., Rosa, R., et al. 2020, A&C, 30, 100334
Bhambra, P., Joachimi, B., & Lahav, O. 2022, MNRAS, 511, 5032
Buta, R. J. 2013, Galaxy Morphology, Planets, Stars and Stellar Systems: Volume 6: Extragalactic Astronomy and Cosmology (Dordrecht: Springer), 1
Calleja, J., & Fuentes, O. 2004, MNRAS, 349, 87
Cheng, T.-Y., Conselice, C. J., Aragon-Salamanca, A., et al. 2020, MNRAS, 493, 4209
de la Rosa, F. L., Gomez-Sirvent, J. L., Morales, R., Sanchez-Reolid, R., & Fernandez-Caballero, A. 2023, Computers & Industrial Engineering, 183, 109549
De Vaucouleurs, G. 1959, Classification and Morphology of External Galaxies, Vol. 5, Astrophysik IV: Sternsysteme / Astrophysics IV: Stellar Systems, Vol. 5 (Berlin: Springer), 275
Dieleman, S., Willett, K. W., & Dambre, J. 2015, MNRAS, 450, 1441
Vavilova, I., Dobrycheva, D., Vasylenko, M., Elyiv, A., Melnyk, O., & Khramtsov, V. 2021, arXiv:1712.08955v1
Draelos, R. L., & Carin, L. 2021, arXiv:2011.08891
Fielding, E., Nyirenda, C. N., & Vaccari, M. 2021, in 2021 Int. Conf. Electrical, Computer and Energy Technologies (ICECET) (Piscataway, NJ: IEEE)
Fukugita, M., Nakamura, O., Okamura, S., et al. 2007, AJ, 134, 579
Gauci, A., Adami, K. Z., & Abela, J. 2010, arXiv:1005.0390
Gavrikov, P., 2020 visualkeras, https://github.com/paulgavrikov/visualkeras
Gunning, D., Stefik, M., Choi, J., et al. 2019, Science Robotics, 4, eaay7120
Hubble, E. P. 1926, ApJ, 64, 321
Iandola, F. N., Han, S., & Moskewicz, M. W. 2016, arXiv:1602.07360
Jeevan, P., & Sethi, A. 2024, arXiv:2406.05612
Kalvankar, S., Pandit, H., & Parwate, P. 2020, arXiv:2008.13611
Katebi, R., Zhou, Y., Chornock, R., & Bunescu, R. 2019, MNRAS, 486, 1539
Khramtsov, V., Vavilova, I., Dobrycheva, D., et al. 2022, arXiv:2209.12194
Lintott, C. J., Schawinski, K., Slosar, A., et al. 2008, MNRAS, 389, 1179
Lotz, J. M., Primack, J., & Madau, P. 2004, AJ, 128, 163
Maltsev, K., Schneider, F., Röpke, F., et al. 2023, A&A, 681, 1
Naim, A., Lahav, O., Sodré, L. J., & Storrie-Lombardi, M. C. 1995, MNRAS, 275, 567
Nair, P. B., & Abraham, R. G. 2010, ApJS, 186, 427
Pasquet, J., Bertin, E., Treyer, M., Arnouts, S., & Fouchez, D. 2018, A&A, 621, 15
Reza, M. 2021, A&C, 37, 1–11100492
Ribeiro, M. T., Singh, S., & Guestrin, C. 2016, in Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (ACM Digital Library), 1135
S Band, S., Yarahmadi, A., Hsu, C.-C., et al. 2023, Informatics in Medicine Unlocked, 40, 101286
Sandage, A. 1961, The Hubble Atlas of Galaxies, 618 (Washington: Carnegie Institution of Washington)
Shapurian, G. 2024, arXiv:2406.11400
Sreejith, S., Pereverzyev Jr, S., Kelvin, L. S., et al. 2017, MNRAS, 474, 5232
Tsivgoulis, M., Papastergiou, T., & Megalooikonomou, V. 2022, Machine Learning with Applications, 10, 100399
Urechiatu, R., & Frincu, M. 2024, Univ, 10, 22
Waghumbare, A., Kasera, S., & Singh, U. 2024, in 2024 3rd Int. Conf. Innovation in Technology (INOCON) (Piscataway, NJ: IEEE)
Weber, P., Carl, K. V., & Hinz, O. 2024, Management Review Quarterly, 74, 867
Wu, D., Zhang, J., Li, X., & Li, H. 2022, RAA, 22, 115011
Yu, Y. 2023, Proc. SPIE, 12566, 125664Z
Zhang, Z., Zou, Z., Li, N., & Chen, Y. 2022, RAA, 22, 055002
Zhao, B., Feng, J., Wu, X., & Yan, S. 2017, International Journal of Automation and Computing, 14, 119
Zhu, X.-P., Dai, J.-M., Bian, C.-J., et al. 2019, Ap&SS, 364, 55