




# LR-Net: A Block-based Convolutional Neural Network for Low-Resolution Image Classification

Ashkan Ganj<sup>1</sup> · Mohsen Ebadpour<sup>2</sup> · Mahdi Darvish<sup>3</sup> · Hamid Bahador<sup>3</sup> 

Received: 6 October 2022 / Accepted: 15 May 2023 / Published online: 27 June 2023  
© The Author(s), under exclusive licence to Shiraz University 2023

## Abstract

The success of convolutional neural network-based architecture on image classification in learning and extracting features has made them very popular recently, but the task of image classification becomes more challenging when we apply state-of-the-art models to classify noisy and low-quality images. It is still difficult for models to extract meaningful features from this type of image due to its low resolution and lack of meaningful global features. Moreover, high-resolution images need more layers to train, which means they take more time and computational power. Our method also addresses the problem of vanishing gradients as the layers become deeper in the deep neural networks that we mentioned earlier. In order to address all of these issues, we developed a novel image classification architecture composed of blocks that are designed to learn both low-level and global features from blurred and noisy low-resolution images. Our design of the blocks was heavily influenced by Residual Connections and Inception modules in order to increase performance and reduce parameter sizes. We also assess our work using the MNIST family datasets, with a particular emphasis on the Oracle-MNIST dataset, which is the most difficult to classify due to its low-quality and noisy images. We have performed in-depth tests that demonstrate that the presented architecture is faster and more accurate than existing cutting-edge convolutional neural networks. Furthermore, due to the unique properties of our model, it can produce a better result with fewer parameters. The source code of the project is available at this GitHub repository.

**Keywords** Image classification · Image processing · Convolutional neural networks · Deep learning

## 1 Introduction

Deep convolutional neural networks (CNNs) have revolutionized computer vision in recent years, and are widely used in various tasks such as object detection [1, 2], image classification [3, 4], and instance image segmentation [2]. CNNs are preferred over feed-forward networks due to their ability to share parameters and reduce dimensionality. In a CNN, features are shared, leading to fewer parameters and therefore reduced computations. The idea behind

CNNs is that pixels and their surroundings carry semantic meaning within an image, and elements of interest can appear anywhere within the image. Like multi-layer perceptrons (MLPs), CNNs have layers, but they are not fully connected. Instead, they have filters, which are sets of weights applied to the whole image.

Since the groundbreaking success of AlexNet [5] in the 2012 ImageNet Large Scale Visual Recognition Competition, which combined CNNs and GPUs, research has focused on improving CNN architecture and integrating new concepts for better performance. VGG [6], GoogleLeNet [3], and ResNet [4] are three popular attempts to improve performance through the use of CNNs. VGG [6] investigates deeper network performance by extracting as many features as possible from high-resolution images, while GoogleLeNet [3] attempts to perform multiple operations with different filter sizes in parallel to reduce the risk of trade-offs. ResNet [4], on the other hand, generates residual learning blocks through identity mapping shortcut

✉ Hamid Bahador  
hamid.bahador@uma.ac.ir

<sup>1</sup> Computer Science, Worcester Polytechnic Institute, Worcester, USA

<sup>2</sup> Department of Computer Engineering, Amirkabir University of technology, Tehran, Iran

<sup>3</sup> Department of Electrical and Computer Engineering, University of Mohaghegh Ardabili, Ardabil, Iran

connections, allowing for neural network models with hundreds or even thousands of layers to overcome the gradient vanishing problem. Other models such as DenseNet [7] have also found that reorganizing connections between layers can improve learning and representation.

While all of these models perform well with high-resolution images, low-resolution images may not always be available due to image age, bandwidth, and computation limitations. Tests with advanced models such as Inception-v3 [8] on low-resolution images such as Oracle-MNIST [9], which contain natural noise, have shown a degradation in performance compared to results with high-resolution images. This problem is not unique to CNNs, as deep neural networks (DNNs) also suffer from it [10]. Many factors influence the quality of an image in the real world. Most of the time, we cannot get a pure and high-resolution image, and most images have some level of noise and degradation, which will diminish the final result. Overall, it has been recognized that poor image quality has a significant impact on the performance of deep neural networks in computer vision tasks, as noted here [11]. So in this paper, we try to directly address this problem with a new architecture by utilizing the idea of inception [3] to get as many features as possible from images by using different kernels and combining them with some residual connections [4] to solve problems like vanishing gradients and the problem of dimensionality in deep neural networks.

This paper commences with introducing some related works in the following section. Next, it explains the approach proposed and presented the experimental setup, and talks about the result of training in sections 3 and 4, respectively, and finally, we have a conclusion in section 5.

## 2 Related Work

The use of convolutional neural networks (CNNs) has been a key method of recognizing images, such as classifying them [5, 12], recognizing actions [13], and locating objects [14]. As deep learning models require a lot of training instances to be able to converge, pre-trained models [15] have been implemented to process small- and medium-sized datasets. There seems to be a noticeable improvement in accuracy with the method mentioned above. Nevertheless, because of the pre-trained weights to large datasets (e.g., ImageNet [16]), it is more time-consuming and computationally intensive than ever. In addition, compared to traditional feed-forward networks, CNNs have the advantage of parameter sharing and dimensionality reduction [17], which leads to a reduction in the number of parameters and computations.

According to Han et al. [18] and Wang et al. [9], in order to challenge deep learning models, they created benchmark

datasets that are acceptable and share the same characteristics of MNIST, namely that the datasets are small in size and encoded in an easy way to use. Images from both Fashion and Oracle are converted into a format compatible with the MNIST dataset. Thus, they can use MNIST's original dataset instantly regardless of the machine learning system. Also, noted datasets contain more complex information than simple digits extracted from the MNIST.

Several innovative models for classifying  $2 \times 28$  images were presented in the literature [19]. In order to characterize the images in the fashion-MNIST dataset using convolutional neural networks, the authors prepared three types of neural networks. The model exhibits amazing results on the benchmark dataset. An extensive correlation was established between various CNN structures (for example, VGG16) on various datasets (for example, ImageNet) using the LeNet-5 network designed for fashion-MNIST. As one example, a custom CNN type with stacked convolution layers of VGG 16 achieved an accuracy rate of 93.07% on the Fashion MNIST in its published study [20]. Various models of CNNs were introduced to determine which of them is most suitable for characterization and identification in terms of their accuracy. The deep learning architectures that were applied were LeNet-5, AlexNet, VGG-16, and ResNet.

In most cases, the models perform exceptionally well on specific data, but they do not generalize well to similar datasets. As an example, [21] proposed a shallow convolutional neural network using batch normalization techniques in order to accelerate training convergence and improve accuracy. The noted network consists of only four layers with small convolution kernels, resulting in a low time and space complexity. Even though the model achieved top accuracy on the digits MNIST dataset [22], it was not able to perform sufficiently on both CIFAR [23] and Fashion MNIST [18].

It is the intention of most of the recently developed deep convolutional neural networks (DCNNs) [24–27] that utilize Inception and Residual connections as the basis to implement bigger deep networks. In order to make the model more accurate and detailed, the parameters of the architecture are increased substantially as the size and depth of the model increases. The complex nature of this training increases the complexity of the model, which, in turn, increases the number of resources required to run it. Recurrence is a difficult property to incorporate in popular Inception architectures, but it is crucial to improving training and testing accuracy by requiring fewer computational resources. Some researchers have attempted to implement more complex DCNN architectures such as GoogleNet [3], or residual networks with 1001 layers [28] that are capable of high recognition accuracy when applied to different benchmark datasets.



As we intend to tackle the problem of handling the low-resolution image and its classification as mentioned in [29, 30], we consider the CNN's first layers as feature extractors. After that, images are classified by taking advantage of these features. In order to maximize efficiency, a custom CNN block is designed. As compared to mainstream DCNN architectures, this model not only guarantees a higher recognition accuracy while requiring fewer computation parameters but also contributes to the overall training process of the deep learning approach as a whole.

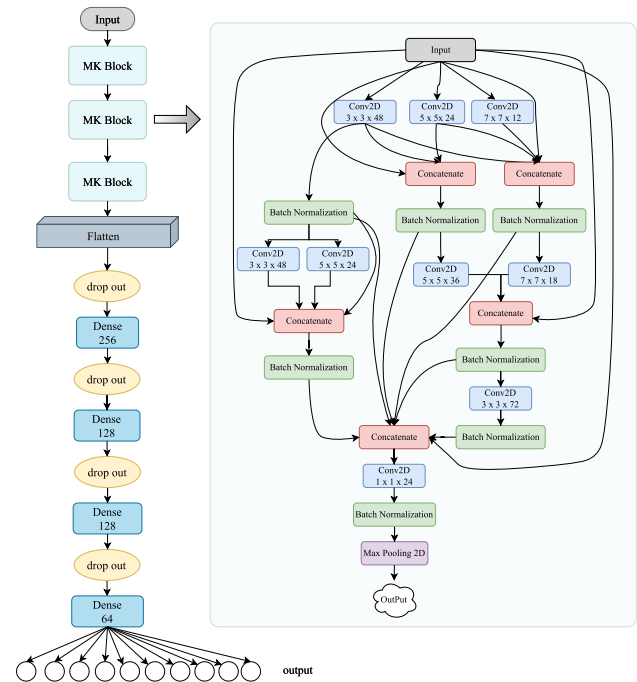
## 3 Method

### 3.1 Problem Overview

In this section, we will discuss the rationale behind our decision to introduce a new architecture for image classification. After doing some tests with some famous state-of-the-art models on the Oracle-MNIST [9], we discover a noticeable decrease in the performance of the models. With some further experiments, we realized that the relation between classification accuracy and the resolution of images is close to being linear, in all the cases, and this problem seems to be common in most image-classification algorithms. Even images with small degradation or noise which is not noticeable by humans can cause performance problems for models. According to our findings, we believe that the primary cause of this problem is the architecture of the models, and in order to solve the issue, we believe that new algorithms and architectures need to be developed. In this paper, we introduce a new architecture for addressing this problem.

### 3.2 Overall Architecture

As one can see from Fig. 1, our architecture consists of three multi-kernel blocks that stack on top of each other, including the steps that our models take to learn the features and details. After that, we add some fully connected layers and the last layer with a sigmoid activation function for classifying the outputs. The proposed architecture uses the concept of Inception and Residual Connections in MK blocks to provide robust performance for classifying images. As we know, the inception modules were introduced as a way of reducing computational expense in CNN, and we knew that the simple models could not solve this problem, so we tried to get the inception module idea and use it with a combination of residual connections to improve accuracy and reduce computation overhead. Since our neural network deals with a wide variety of images



**Fig. 1** An illustration of the overall architecture of the model is shown in this figure, along with details on each of the blocks. Our block architecture is influenced by both Inception and Residual Net concepts. The block can be divided into two different sides, from which we can extract specific features. As part of our approach, three kernels ( $3 \times 3$ ,  $5 \times 5$ ,  $7 \times 7$ ) were used for feature extraction and a  $1 \times 1$  kernel was used for feature combining at the last step

with varying salient parts or features, so using an inception base idea is essential for this architecture.

As our input images are low-resolution images, the useful details usually exist in fewer pixels, and for each window or kernel, the information that can be extracted is rarely found, so the filter size in our models is descending, which means that for larger sizes, we will have smaller kernel sizes. Additionally, the padding of each convolutional layer is the same since the results must be concatenated.

The multi-kernel block (MK-block) contains several residual connections, as shown in Fig. 1. Rather than learning unreferenced functions, these links learn residual functions by referencing the layer inputs. The stacked nonlinear layers have the ability to skip connectivity to fit another mapping of  $F(x) := H(x) \times x$  that corresponds to the desired underlying mapping  $H(x)$ .  $F(x) + x$  is formed from the initial mapping (see Fig. 2). The residual mapping is generally easier to tweak than the original one. In theory, fitting an identity mapping by a stack of nonlinear functions requires less effort than pushing the residual to zero if an identity mapping is optimal.

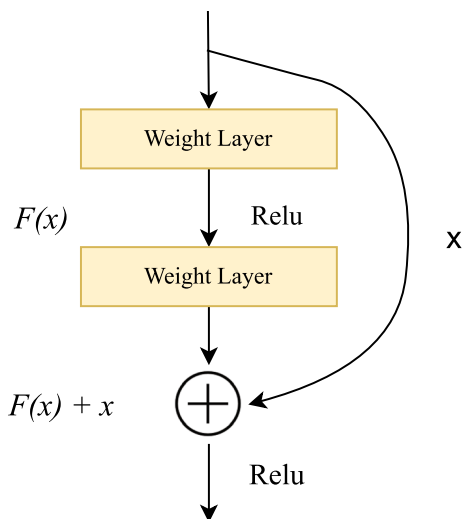


Fig. 2 Building blocks of the deep residual network

### 3.3 Multi-Kernel Block

As we mentioned earlier, our model was constructed by stacking three multi-kernel blocks (Mk-kernel) on top of each other. Our architecture relies heavily on these identical blocks. At the first layer (Fig. 3) of our block, we have three different kernels ( $3 \times 3$ ,  $5 \times 5$ ,  $7 \times 7$ ). The connections of these three kernels is critical because we want to share the information in a way that our model does not experience a big jump in kernel and feature. In other words, we connect the result of kernels that have meaningful relationships between them. For example, we have a connection between kernels  $3 \times 3$  and  $5 \times 5$  because the information that they exchange is useful. However, we do not have a connection between  $3 \times 3$  and  $7 \times 7$  because the information that they will share is unusable. The second reason to have this type of connection is that we want to process the low-resolution information. This is because as we know, the big kernels extract more global information,

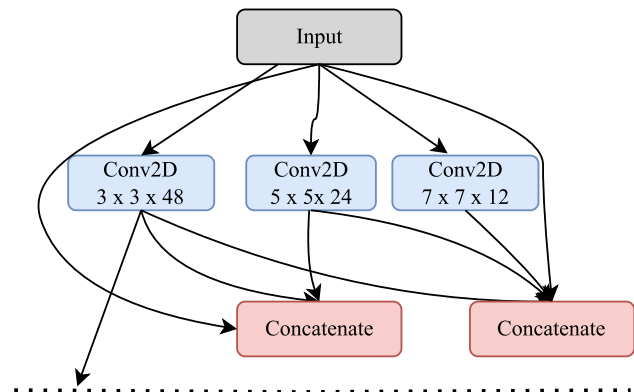


Fig. 3 MK-block’s first layer shows details of kernel connections as well as how information is shared between them

while smaller kernels extract detail and local information. If you look at Fig. 1, you will see that we have a connection between  $3 \times 3$ ,  $5 \times 5$ , and  $7 \times 7$ . The reason we made this connection is that the  $5 \times 5$  kernel can create a balance between the  $3 \times 3$  and  $7 \times 7$  images in a way that the model will better understand which parts of the image will have a local view and which parts will have a global view.

We divide our block into two parts in the second layer. On the right side, we connect the nodes with kernel  $5 \times 5$  and kernel  $7 \times 7$  together, and at this step, the model attempts to conclude the information in big kernels without considering details, and it will use this information at the last layer when we want to aggregate all the features. Our next layer is a  $3 \times 3$  Conv layer that gathers the last details from small sizes, or we can say that we will do local features extraction or low-level features extraction based on global features. The model will try to extract as much detail and low-level features as it can in the continuation of conv2D with kernel  $3 \times 3$ , since in the previous layer we extracted some low-level features, and so we repeat this operation again. The reason we need this step is that we need good performance on noisy and blurry images. At the end of the process, the extracted features should be applied with any weight from the previous steps.

The last point to note is the way we chose the number of filters. In our images, we have two types of features. Some of them are local and some of them are global features. When we choose small filters, the model will extract the small and detailed features, which will perform poorly with low-quality and noisy images, while when we choose larger kernels, a much more complex neural network will be needed to extract all useful images. As a result, we decided to implement the inception concept, which uses different image sizes across layers because our goal is to classify the low-resolution images. So, we decided to have more  $3 \times 3$  kernels than  $7 \times 7$  and  $5 \times 5$  kernels.

## 4 Experiments

### 4.1 Datasets

We use MNIST-family datasets to evaluate our work results and compare them with other state-of-the-art models to show the superiority of our model. The MNIST family datasets have both of the characteristics we want to study, which are noise and low-quality images. Table 1 shows a summary of all the information provided in the next parts.

**Table 1** Information regarding the size and number of instances of datasets used for evaluation and training in this paper

Dataset name	Training data		Validation data		Test data	
	Instances	Size	Instances	Size	Instances	Size
Digit MNIST	–	64.73	–	88.41	–	94.13
Fashion MNIST [18]	71.1	63.32	36.4	88.70	20.7	94.93
Oracle MNIST [9]	49.4	68.55	22.3	91.73	14.1	96.28

#### 4.1.1 MNIST Digit Dataset

The fact that most of the models use this dataset and that it also meets our requirement for a low-resolution image led us to choose it as a way to demonstrate the power of our architecture while providing a good way to compare our results with those of other state-of-the-art models. The MNIST dataset was created using NIST's Special Databases 3 and 1, which include binary images of handwritten integers. Initially, NIST classified SD-3 as the training set and SD-1 as the test set. SD-3, on the other hand, is much cleaner and easier to discern than SD-1. The MNIST training set contains 30,000 SD-3 patterns and 30,000 SD-1 patterns. Our test set included 5000 SD-3 patterns and 5000 SD-1 patterns. The SD-1 has 58,527 digit images authored by 500 different writers. In contrast to SD-3, where blocks of data from each writer appear in sequence, the data in SD-1 is fragmented.

#### 4.1.2 Fashion MNIST Dataset

Fashion-MNIST [18] is a dataset of Zalando article photos, with 60,000 examples in the training set and 10,000 examples in the test set. Each example is a 28 x 28 grayscale image paired with a label from one of ten categories. Fashion-MNIST is intended to be a drop-in replacement for the original MNIST dataset for evaluating machine learning algorithms. The image size and structure are the same as in the training and testing splits.

Each image is 28 pixels high and 28 pixels wide, for a total of 784 pixels. Each pixel has a single pixel value associated with it, which indicates how light or dark that pixel is, with larger numbers indicating darker. This pixel value is an integer ranging from 0 to 255. There are 785 columns in the training and test data sets. The class labels are listed in the first column. The first represents the clothing, while the second represents the accessories. The remaining columns contain the corresponding image's pixel values.

#### 4.1.3 Oracle MNIST Dataset

The Oracle-MNIST dataset [9] contains 30,222 ancient characters from ten categories in 28 x 28 grayscale image format for pattern classification, with special challenges in

image noise and distortion. The training set has 27,222 images, while the exam set has 300 images per class. It uses the same data structure as the original MNIST dataset, making it compatible with all existing classifiers and systems. However, it is more difficult to train a classification model on it without overfitting and low performance. Images of ancient characters suffer from (1) incredibly serious and unusual noises created by 3000 years of burial and aging, as well as (2) significantly different writing styles in ancient Chinese, both of which make them realistic for machine learning study.

We chose this dataset because of its noisy and low-resolution characteristics. This aspect of the dataset makes it extremely difficult for a standard model to classify the images. Figures x and y show that the Inception v3 [8] and Vgg-16 [6] models are not performing optimally, and we can see an obvious degradation in the models' performance.

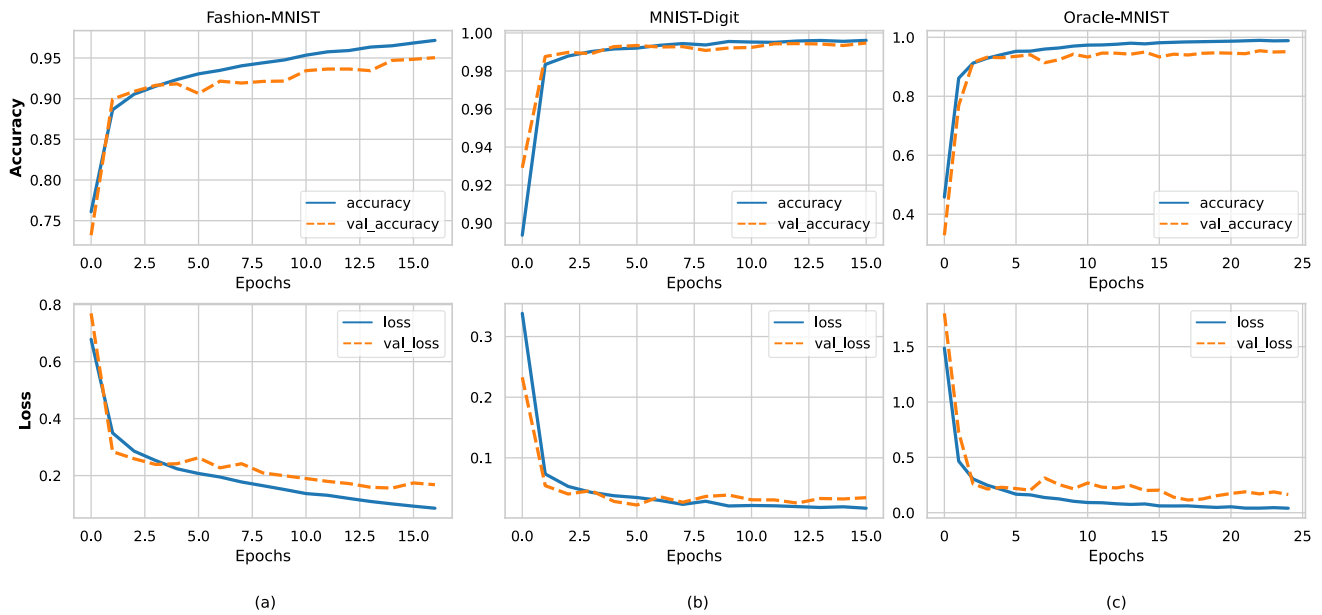
#### 4.2 Training Setup

To improve the reliability of our results, we use the same setup and input size for training and testing all models. The images have been resized to 35 x 35 x 1 in order to preserve the standard input size for models like Inception-v3 [8] and VGG-16 [6], which do not function properly with 28 x 28. For training the models, we use Google-Colab with an NVIDIA Tesla T4 GPU with 16GB memory, with a batch size of 128. For analyzing the model's behavior over a long period of time, models were trained for 200 epochs. However, for publishing the model, we use a callback function that stops the training after 30 unchanged epochs on validation loss. This technique is applied to save time and prevent overfitting.

#### 4.3 Results

In this section, we will present the results of our model and we will have a comparison of our model's accuracy and loss against other models to evaluate the effectiveness of the introduced architecture in low-resolution image classification tasks.

As shown in Fig. 4, our model outperformed three other state-of-the-art models in terms of accuracy on all three datasets without experiencing gradient exploding or



**Fig. 4** Training and validation loss over epochs generated by training the custom CNN model on **a** MNIST fashion, **b** digit MNIST, and **c** oracle MNIST datasets. The gradual increment in accuracy suggests

the efficiency of the network for learning practical features of the image with an optimal convergence pace

overfitting issues. It is important to note that the introduced model is simpler and faster based on the number of trainable and un-trainable parameters and the prediction time. Table 2 provides a detailed comparison of these five models on the Oracle dataset [9]. These better results were achieved by utilizing the MK-Blocks introduced in Sect. 3. Our analysis of Tables 2 and 3 shows that noise and details in the images directly affect all models' performance. Due to a large amount of natural noise in the Oracle-MNIST dataset [9], as well as internal variance in the labels, the accuracy of the models is lower on this dataset. However, the introduced method (LR-Net) is able to achieve an accuracy of 95.13, which is higher than other state-of-the-art models that we examined in this research. We achieved higher accuracy with fewer parameters and a shorter prediction time, which shows the effectiveness of LR-Net in classifying low-resolution images. The results we discussed earlier were expected as a result of the unique block-based architecture we introduced in Sect. 3. Each

block was maintained by using a number of residual connections between the input and upper levels in order to maintain the features. This proved very effective at increasing the meaningful feature extraction of models. We were also able to overcome the gradient explosion problem, which is common in deep neural networks. Based on the results, we can see the power of residual connections in combination with inception modules.

#### 4.4 Comparative Evaluation

The purpose of this section is to compare our method with VGG-16 [6], Inception-V3 [8], AlexNet [5], and ResNet50 [4] on three popular datasets: MNIST-Digit, MNIST-Fashion [18], and Oracle-MNIST [9]. In Sect. 3, we discussed the use of the Inception module and the Res connection to improve the accuracy and performance of the LR-Net model on low-resolution images. Therefore, it may

**Table 2** Analysis of five different models using the Oracle dataset and their parameters and prediction and running times

Model name	Prediction time	Number of trainable parameters	Number of non-trainable parameters	Number of parameters
AlexNet	24 ms	56,361,738	2752	56,358,986
Inception-V3	22 ms	23,851,784	34,432	23,817,352
VGG-16	16 ms	138,357,544	0	138,357,544
ResNet-50	15 ms	25,636,712	53,120	25,583,592
Ours(LR-Net)	10 ms	1,028,234	3236	1,024,998

**Table 3** The accuracy of five different image classification algorithms on three famous MNIST datasets

Model name	Digit MNIST	Fashion MNIST [18]	Oracle MNIST [9]
AlexNet	98.4	91	85.7
Inception-V3	93.31	94.44	92.6
VGG-16	92.4	74.2	52.6
ResNet50	98.8	88.63	90.9
Ours	99.47	95.03	95.13

be worthwhile to compare these two models with others to see how they perform.

As shown in Table 3, our newly developed model has higher accuracy than other models. This is achieved by utilizing MK-Block, which also reduced the number of parameters and prediction time, as shown in the Table 2. This reduces computation overhead and increases prediction speed, which is particularly effective for low-power devices. With 99.47% accuracy, our model is more accurate than other models in the MNIST dataset. In the second dataset (Fashion-MNIST [18]), which is a more challenging dataset to classify due to image complexity, our model can achieve better accuracy compared to others by looking at Table 3. We can see that the model keeps its robustness during training.

The real challenge comes with Oracle-MNIST [9], since it is a newly collected dataset with unique characteristics. As indicated in Sect. 4.1, there are some natural noises and blur that come with images. As a result, the images have very poor quality, which will reduce the performance of the models and make it harder for models to extract features. Based on our tests, which are available in Tables 3 and 2, other state-of-the-art models (inception-v3 [8], VGG-16 [6], AlexNet [5], ResNet50 [31]) had very poor performance on Oracle-MNIST [9], despite their higher number of parameters and complexity of their architecture (Table 2). However, our model achieves a better result on this dataset and it is able to classify images with greater confidence while having less power consumption than Inception-v3 [8] and VGG-16 [6].

#### 4.5 Significance Test

The purpose of this section is to demonstrate the robustness of our results and model by analyzing our testing results statistically. In order to ensure that weight sharing did not happen between training sessions, we started training the models from the beginning at isolated run times. This was to increase the reliability of our results.

Table 4 illustrates the information of the training model ten different times on the Oracle-MNIST [9] dataset, which was our main goal. During all training runs, we made sure that the model learns weights from the beginning without using transferring learning from previous runs. So, the

**Table 4** The results for ten different runs of LR-Net on Oracle-MNIST [9]

	Standard deviation	Mean	Variance	Mode
Results	0.0030	0.9518	9.4140	0.9549

model's results are independent of training status. The results of Table 4 depict the robustness of our model and show its reliability in the real world. By looking at the table, we can see the average validation accuracy which is 95.18%, which is around the value that we reported earlier in Table 3. The variance and standard deviation show our error rate in relation to what we announced. According to Table 4, the results and values announced in this paper are reliable and applicable to other environments.

## 5 Conclusions and Future Work

In this paper, a novel architecture for low-resolution image classification is proposed and examined. Using the results of our study, we can conclude that this model can outperform many of the state-of-the-art models that are currently available in image classification tasks. In addition, the presence of modules similar to those seen at inception may contribute to these results. With the help of these two ideas, we were able to create a model that was simpler and more efficient compared to others. The model was evaluated on MNIST family datasets and is generalizable to other low-resolution ones.

As a future work, there are several modifications that can be performed to make our model more robust against noises and make it faster by reducing the number of parameters. According to our findings, the image size decreased dramatically after the first MK block, so having the same number of filters for the second and third MK blocks would not be optimal, which increases computational costs and would be undesirable. The goal is to introduce a new hyperparameter for the number of filters in the second and third blocks in order to make these blocks more flexible for the individual image. Another way to manage this problem is to use deconvolution to increase the

image size after each block. This will increase feature extraction and make the system more robust overall.

## References

- Mazzia V, Khaliq A, Salvetti F, Chiaberge M (2020) Real-time apple detection system using embedded systems with hardware accelerators: an Edge AI application. *IEEE Access* 8:9102–9114
- He K, Gkioxari G, Dollár P, Girshick R (2017) Mask R-CNN. In: 2017 IEEE international conference on computer vision (ICCV), pp 2980–2988
- Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A (2015) Going deeper with convolutions. In: 2015 IEEE conference on computer vision and pattern recognition (CVPR), pp 1–9
- He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: 2016 IEEE conference on computer vision and pattern recognition (CVPR), pp 770–778
- Krizhevsky A, Sutskever I, Hinton GE (2012) ImageNet classification with deep convolutional neural networks. In: Proceedings of the 25th international conference on neural information processing systems—Volume 1, NIPS’12, (Red Hook, NY, USA), pp 1097–1105, Curran Associates Inc.
- Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*
- Huang G, Liu Z, Van Der Maaten L, Weinberger KQ (2017) Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4700–4708
- Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z (2016) Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2818–2826
- Wang M, Deng W (2022) Oracle-MNIST: a realistic image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:2205.09442* pp 1–7
- Dodge S, Karam L (2016) Understanding how image quality affects deep neural networks. In: 2016 eighth international conference on quality of multimedia experience (QoMEX), pp 1–6, IEEE
- Koziarski M, Cyganek B (2018) Impact of low resolution on image recognition with deep neural networks: an experimental study. *Int J Appl Math Comput Sci* 28:735–744
- Nath SS, Mishra G, Kar J, Chakraborty S, Dey N (2014) A survey of image classification methods and techniques. In: 2014 international conference on control, instrumentation, communication and computational technologies (ICCICCT), pp 554–557
- Gkioxari G, Girshick R, Malik J (2015) Contextual action recognition with R\* CNN. In: Proceedings of the IEEE international conference on computer vision, pp 1080–1088
- Bell S, Zitnick CL, Bala K, Girshick R (2016) Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2874–2883
- Zhuang F, Qi Z, Duan K, Xi D, Zhu Y, Zhu H, Xiong H, He Q (2019) A comprehensive survey on transfer learning
- Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L (2009) ImageNet: a large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition, pp 248–255, IEEE
- Sorzano COS, Vargas J, Montano AP (2014) A survey of dimensionality reduction techniques. *arXiv preprint arXiv:1403.2877*
- Xiao H, Rasul K, Vollgraf R (2017) Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms
- Hirata D, Takahashi N (2020) Ensemble learning in cnn augmented with fully connected subnetworks. *arXiv preprint arXiv:2003.08562*
- Greeshma K, Sreekumar K (2019) Hyperparameter optimization and regularization on Fashion-MNIST classification. *Int J Recent Technol Eng (IJRTE)* 8(2):3713–3719
- Lei F, Liu X, Dai Q, Ling BW-K (2020) Shallow convolutional neural network for image classification. *SN Appl Sci* 2(1):1–8
- Deng L (2012) The MNIST database of handwritten digit images for machine learning research. *IEEE Signal Process Mag* 29(6):141–142
- Krizhevsky A, Sutskever I, Hinton GE (2012) ImageNet classification with deep convolutional neural networks. *Adv Neural Inf Process Syst*, 25
- Wang W, Yang Y, Wang X, Wang W, Li J (2019) Development of convolutional neural network and its application in image classification: a survey. *Opt Eng* 58(4):1–19
- Rawat W, Wang Z (2017) Deep convolutional neural networks for image classification: a comprehensive review. *Neural Comput* 29(9):2352–2449
- Aloysius N, Geetha M (2017) A review on deep convolutional neural networks. In: 2017 international conference on communication and signal processing (ICCSP), pp 0588–0592, IEEE
- Chen T, Kornblith S, Norouzi M, Hinton G (2020) A simple framework for contrastive learning of visual representations. In: International conference on machine learning, pp 1597–1607, PMLR
- He K, Zhang X, Ren S, Sun J (2016) Identity mappings in deep residual networks. In: European conference on computer vision, pp 630–645, Springer
- Wu Y, Zhang Z, Wang G (2019) Unsupervised deep feature transfer for low resolution image classification. In: Proceedings of the IEEE/CVF international conference on computer vision workshops, pp 0–0
- Zhu X, Li Z, Li X, Li S, Dai F (2020) Attention-aware perceptual enhancement nets for low-resolution image classification. *Inf Sci* 515:233–247
- He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.