Biomedical Engineering Letters
https://doi.org/10.1007/s13534-020-00162-9

ORIGINAL ARTICLE

Synthetic image augmentation with generative adversarial network for enhanced performance in protein classification

Rohit Verma[1] · Raj Mehrotra[1] · Chinmay Rane[1] · Ritu Tiwari[1] · Arun Kumar Agariya[1]

Received: 12 November 2019 / Revised: 2 May 2020 / Accepted: 29 June 2020

Abstract
Proteins are complex macromolecules accountable for the biological processes in the cell. In biomedical research, the images of protein are extensively used in medicine. The rate at which these images are produced makes it difficult to evaluate them manually and hence there exists a need to automate the system. The quality of images is still a major issue. In this paper, we present the use of different image enhancement techniques that improves the contrast of these images. Besides the quality of images, the challenge of gathering such datasets in the field of medicine persists. We use generative adversarial networks for generating synthetic samples to ameliorate the results of CNN. The performance of the synthetic data augmentation was compared with the classic data augmentation on the classification task, an increase of 2.7% in Macro F1 and 2.64% in Micro F1 score was observed. Our best results were obtained by the pretrained Inception V4 model that gave a fivefold cross-validated macro F1 of 0.603. The achieved results are contrasted with the existing work and comparisons show that the proposed method outperformed.

Keywords Image enhancement · Generative adversarial network · Protein Image classification · Convolutional neural network · Transfer learning

1 Introduction

Spatial partitioning of functions related to life is an essential activity and forms the basis for life. In the human cell, proteins play a very vital role in making numerous functions happen that jointly facilitate life. Proteins belong to the molecules that are complex and big (composed of hundreds or thousands of smaller amino acids forming a big chain) at the same time and also stand obligatory for the essential function, structure and proper management of the tissues and organs of the body. Normally in biomedical research, images containing proteins in cells are utilized. A leap forward is being held by these cells in the field of medication. Nonetheless, keeping the advances made by high-throughput microscopy in mind, these pictures are produced at a far more rapid speed than what can be physically assessed.

Most of the precise annotations in the medical stream are time-consuming. A much-needed requirement for improved computerization and automation of biomedical picture analysis exists to fasten the comprehension of disease and human cells. Historically, only single patterns were subject to classification, but to perceive the entire complexity of the human cells, there is a requirement of models that could classify even a mixture of patterns. Earlier, Support Vector Machines (SVM), k-NN classifiers, Decision Trees and Artificial Neural Networks (ANN) were the methods developed as an attempt to make the classification of protein images automated. In recent times, the classification of protein localization of single localizing proteins in human cells [1] and budding yeast [2, 3] have been successfully exercised by deep Convolutional Neural Networks (CNNs). Deep CNNs are a specific class of models that alternatively apply trainable filters and local neighborhood pooling operations on the input images to gradually extract more complex features. Many recent attempts have been made by research scholars to apply them in biological domains [4, 5] and to determine the subcellular localization of protein [3] [6, 7].

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s13534-020-00162-9) contains supplementary material, which is available to authorized users.

✉ Rohit Verma
rohitmerc7@gmail.com

[1] Soft Computing and Expert Systems Laboratory, ABV-IIITM, Gwalior, M.P. 474015, India

Published online: 13 July 2020

Various standard pre-trained deep CNN models like VGG16 [8], ResNet [9], InceptionV4 [10], DenseNet [11] and NASNet [12] have been developed and successfully applied to applications related to medical imaging [13, 14] and computer vision [15–17].

The performance of these models is limited by the quality of images, especially in biomedical research where the captured images often have poor contrast and noise. Thus, there exists a greater urge for deploying image enhancement techniques, before utilizing the models. Image Enhancement techniques like Histogram Equalization (HE), Adaptive Histogram Equalization (AHE) [18] and Contrast Limited Adaptive Histogram Equalization (CLAHE) [19] were used to enhance the contrast of the images.

In many cases, existing unusual and scarce cellular structures give birth to extreme class imbalance which raises challenges for a classifier to predict labels accurately. Researchers try to tackle this issue through the use of data augmentation. In the standard procedure in computer vision tasks, an improvement in the training process has been demonstrated using classical data augmentation [20]. A more sophisticated form of augmentation is the use of synthetic data examples being generated by Generative Adversarial Networks (GANs) which enable greater variability and enrichment to the dataset, thereby enhancing the training process [21]. GANs have achieved considerable popularity in the computer vision and related fields. In recent times, a couple of variations of GANs were proposed for producing realistic natural images [22–25]. Recently, a lot of applications based on medical imaging have incorporated the GAN framework [26–32]. Segmentation of images of the heart and lung fields from chest X-ray images were performed by GAN trained by Dai et al. [27]. Many recent papers focused on ameliorating the stability of training and the resulting visual quality of GAN generated images [33–35]. Techniques such as feature matching, minibatch features, virtual batch normalization, minibatch discrimination and one-sided label smoothing have been proposed for improving the stability of GAN [36]. They used some of the innovations regarding the architecture of "Deep Convolutional GAN" (DCGAN) proposed in Radford et al. [22].

In our work, we present an approach for the classification of the protein images. We demonstrate the use of diverse image enhancement techniques on the protein images. The classification of these images is then performed using the pre-trained models. The study also highlights the use of generative models for data augmentation. and compares the same with the classical data augmentation methods. To the best of our knowledge, the generative models haven't been explored on our dataset. In this experiment, we explored variants of GAN such as Conditional GAN, DCGAN for augmentation. Amongst these architectures, the DCGAN was the most stable in terms of training and gave remarkable results in terms of images' quality. Lastly, to conclude, we present a comparison with the existing literature.

2 Data

The dataset images as shown in Fig. 1 have been taken from the Human Protein Atlas Image Classification Kaggle competition [37]. It comprises 31,072 microscopic protein images, each having dimensions of 512 × 512 pixels. There are 28 target classes of proteins and an image may contain more than one type of protein hence making it a multi-label classification problem. All image samples are represented by four filters, the protein of interest (green) plus three cellular landmarks: nucleus (blue), microtubules (red), endoplasmic reticulum (yellow). The green filter should hence be used to predict the label, and the other filters are used as references.

Also, we have used external data having 75,040 samples with red, green and blue filters [38]. We decided to drop the yellow filter because a lot of images didn't have the filter

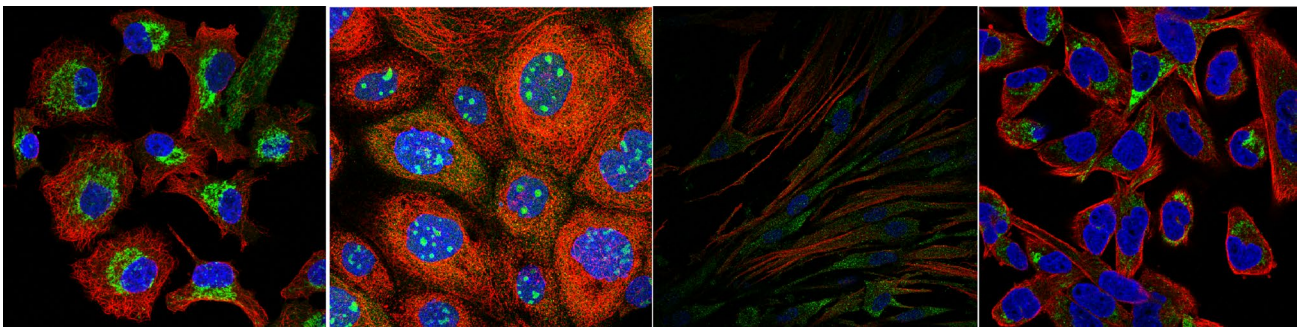

Fig. 1 Sample images

in the external data and also to avoid extra computation on more than 1 lakh images.

3 Methodology

The broad sections of the methodology include use of image processing and enhancement techniques to improve the quality of image, multilabel image classification the pre-trained models by the approach of transfer learning, evaluation of Generative Adversarial Networks (GANs) for synthetic data augmentation and its comparison with traditional data augmentation techniques. The entire methodology is illustrated in the form of a flowchart in Fig. 2.

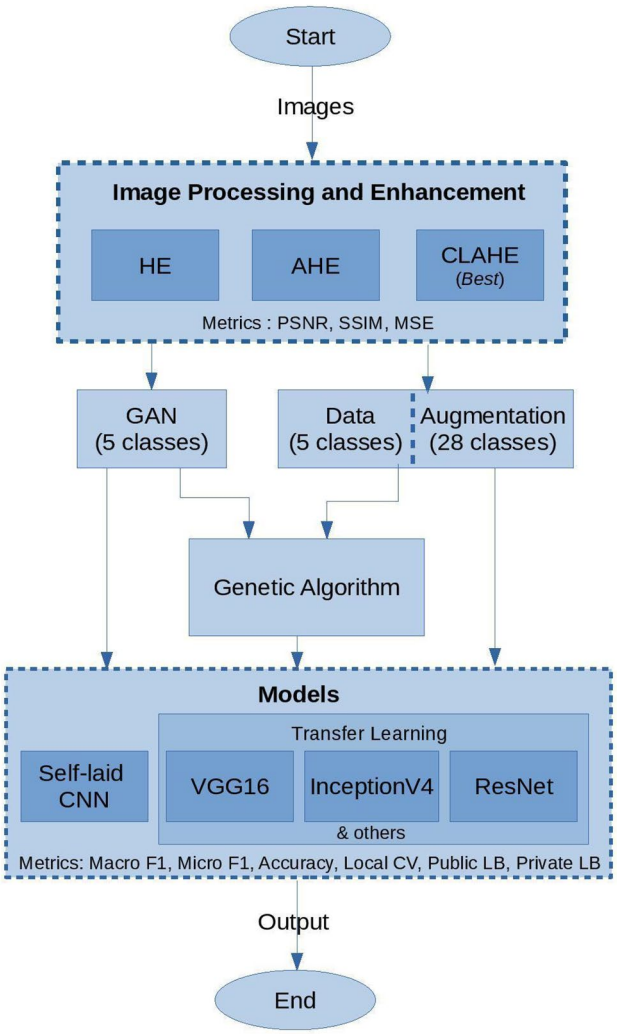

Fig. 2 Methodology flowchart

3.1 Pre-processing

3.1.1 Image enhancement

Contrast is a significant factor while assessing an image's quality. It refers to the difference in the visual properties of an object that distinguishes it from other objects and the background. In visual terms, contrast is resolved by the change in the brightness and colour of the object with other objects. As a result, a higher contrast causes the features to become more prominent than before and thereby strengthening the classification model.

The original images were deprived of contrast and considering contrast enhancement seems to be a reasonable option. This section presents three techniques for contrast enhancement - Histogram Equalization (HE), Adaptive Histogram Equalization (AHE) and Contrast Limited Adaptive Histogram Equalization (CLAHE).

1 HE is a technique in image processing that enhances the contrast of the image by adjusting the corresponding intensity values of the image. It alters the intensity values such that the histogram of the resulting image maps to a specific histogram.
2 AHE is an image processing technique that is built over the limitations of HE. It helps in improving local contrast and sharpening the edges in each region of the image.
3 CLAHE is an improvement over AHE since AHE has a flaw of overamplifying the noise. The HE that we previously defined, considers the global contrast of the image while CLAHE enhances the local contrast of an image.

For our task, the CLAHE technique gave the best performance. We have therefore used images enhanced by CLAHE throughout our experiment.

3.1.2 Generating protein images using augmentation

The key task is to assess the classification performance by using synthetic data augmentation. The effectiveness of GANs for data augmentation is then evaluated and compared with the traditional methods. The data augmentation was thus undertaken in two ways: (1) Traditional augmentation that includes a range of conventional image transformations; (2) Synthesizing new samples learned using a generative network. A brief overview of standard data augmentation techniques is first presented which is followed by the methods used to generate synthetic samples using GANs.

3.1.2.1 Traditional data augmentation Conventional augmentation consists of using a blend of affine transforma-

tions to augment the training data. The transformations used include horizontal flip, vertical flip, crop, scaling, rotation, shear, sharpening, and inversion. Further to ensure more variation, we set a probability value in each of the aforementioned to ensure that not all images are transformed using the same group of transformations, thus diversifying the pipeline for each image. This facilitated more generalization to our models and prevented overfitting.

3.1.2.2 Generative adversarial networks (GAN) for protein image synthesis GANs [31] are a special framework of generative models. The purpose of a generative model is to essentially grasp the inherent distribution of the data p_{data} from a set of examples x (1),…, x(m) to create synthetic images from the learned distribution. The GAN architecture comprises of two neural networks that are trained in an alternating manner. The first network is denoted by D and is called the discriminator which distinguishes between the real and the fake images. A sample x is fed as an input to the network and it outputs D(x), the probability of a sample being a real one. The second network denoted by G is the generator that generates new images that D will regard as real with high probability. G takes random noise samples z(1),…,z(m) as input derived from a uniform or normal distribution p_z and up samples it to form the image G(z) closely resembling the original ones i.e. $p_g = p_{data}$.

We have used a variant of GAN called Deep Convolutional GAN (DCGAN). A typical DCGAN architecture is shown in Fig. 3. In DCGAN both generator (G) and discriminator (D) networks comprise of deep CNN architectures.

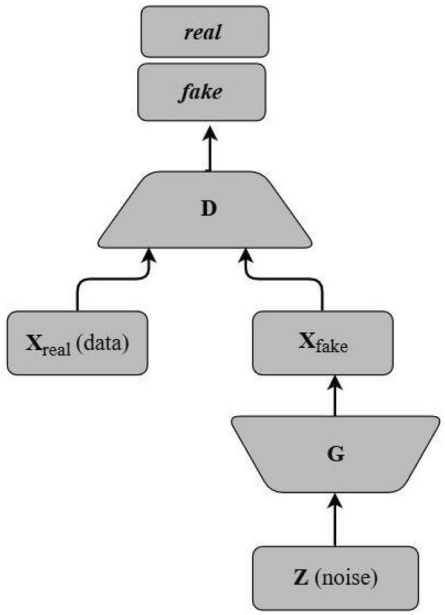

Fig. 3 DCGAN architecture

3.1.2.3 Generator architecture The generator network takes input as a noise vector of 100 random numbers following a Gaussian distribution and converts it to an image of dimensions same as that of the real image i.e. 512 × 512 in our case. Figure 4 shows the generator and discriminator architecture used. The architecture consists of a dense layer reshaped to the size of 8 × 8 × 1024 and six deconvolution layers to up-sample the image with a 3 × 3 kernel size. In between the deconvolution layers, a batch normalization layer has been used with a momentum rate of 0.80. A ReLU activation function is used with the deconvolutional layers and a tanh activation function is used in the final layer.

3.1.2.4 Discriminator architecture The discriminator works on images of size 512 × 512 and outputs a probability of the image being real. The architecture consists of six convolutional layers with a 3 × 3 kernel size. The final layer is a fully connected layer with a single neuron. Strided convolutions are applied in each convolution layer for downsampling. Batch normalization layer and a dropout layer with a dropout rate of 0.25 are added between convolutional layers. A LeakyReLU activation function is used with the convolutional layers and a sigmoid activation function is used after the last fully-connected layer.

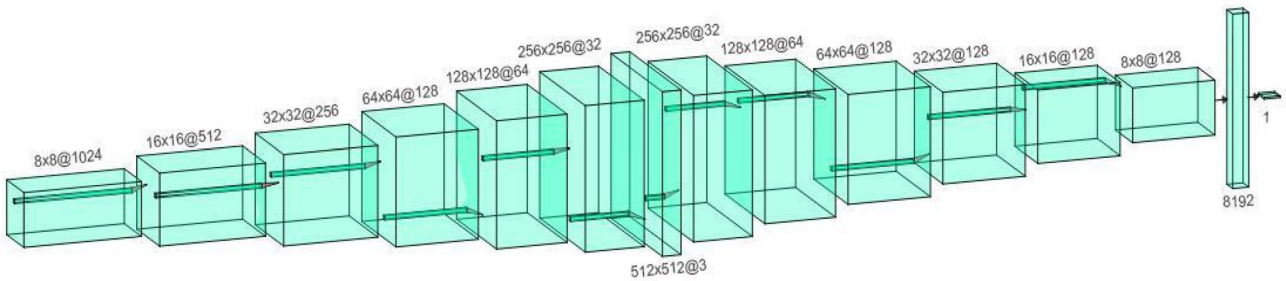

Fig. 4 Schema of the generative adversarial network architecture used

3.1.3 Training procedure for deep convolutional GAN (DCGAN)

The training of GAN's is defined as a two-player minimax game which tries to optimize the following cost function as shown in Eq. (1):

$$min_G max_D V(D,G) = E_{x\sim p_{data(x)}}\left[log(D(x))\right] + E_{z\sim p_z(z)}\left[log(1-D(G(z)))\right] \quad (1)$$

The DCGAN was trained separately in an isolated fashion for each of the protein classes. Once the distribution of each protein class is learned, the synthesis of new images can be accomplished using a normally distributed noise as an input vector. We decided to augment 1000 synthetic images for each class to increase the training data.

The training of discriminator is done in such a way so as to minimize D(x) for samples with $x \nsim p_{data}$ while maximizing D(x) for samples with $x \sim p_{data}$. The task of the generator is to fool D by producing samples G(z) such that $D(G(z)) \sim p_{data}$. This results in maximization of D(G(z)) and correspondingly minimizing $1 - D(G(z))$. Gradually, the generator becomes better at synthesizing realistic images while at the same time discriminator progressively ameliorates the ability to differentiate the real samples from fake ones. Hence the name - adversarial training. The authors of the DCGAN paper have also provided some guidelines for stable training of GAN under most circumstances [22].

Steps for training GAN

1 Normalize images in the range [− 1,1].
2 Use the last activation function in the generator model as tanh.
3 Use ADAM optimizer for generator and SGD for the discriminator.
4 Flip 5% of labels i.e. real becomes fake and fake becomes real, when training discriminator.
5 Construct separate mini-batches for fake and real images.
6 Use of Conv2D + stride, Average Pooling for downsampling and ConvTranspose2D + stride for upsampling.
7 Use label smoothing when training the discriminator.

3.2 Multilabel image classification

In multi-class classification, every image map to a single target label whereas in a multilabel image classification problem, every image may possess more than one label. The entire dataset consists of images belonging to 28 classes. We aimed to evaluate the performance of GAN for synthetic data augmentation on 28 classes but that gave suboptimal performance because not all of the classes possessed ample images for GAN to be trained and therefore, we had to select a subset of these 28 classes for this purpose. Hence in our experiment, in addition to performing classification on 28 classes, we chose a subset of the top 5 most populous classes to produce their synthetic images. These 5 classes account for nearly 56% of the original data. The classification is then done on this selected subset both before and after synthetic data augmentation.

The following section presents the methodology used for performing classification on images of proteins into these 28 classes.

3.3 Classification on 28 classes

3.3.1 Models and validation scheme used

A fivefold cross-validation scheme is used. The training and validation set consist of 84,890 images and 21,222 images respectively. These pre-processed images were then fed into the DenseNet 121, InceptionV3 / V4, Xception, Resnet 50 / 101 and NASNet Mobile architectures. Also, for each of the models, we have tried two image sizes, one of 256 and the other of 512 pixels. A general trend evident from the result is that the larger the dimension of the image, the better the score is. The best results were obtained for the InceptionV4 model on an image size of 512 X 512 pixels.

3.3.2 The best architecture on 28 classes

The architecture used on the top of the InceptionV4 model was a dropout layer of 0.5 rate succeeded by a batch normalization layer, global average pooling layer, dense layer of 128 neurons with an activation function as ReLU and finally a dense layer of 28 neurons with a sigmoid activation function. We trained the model for 35 epochs using Adam as an optimizer with an initial learning rate of 0.0005 and loss function as focal loss. Focal loss is a cross-entropy loss in which the contribution of each sample is added to the loss based on the classification error generated. The objective is that contribution to the loss decreases for every sample being correctly classified by the CNN. Equation (2) shows the focal loss defined as defined in [39].

$$FL = -\alpha_t(1-p_t)^{\gamma} log\left(p_t\right) \quad (2)$$

p_t has to be the ground truth probability of the sample being as class i, γ is a focusing parameter having a prefixed positive scalar value and

$$\alpha_t = \{^{\alpha if y=1}_{1-\alpha otherwise} \quad (3)$$

where α is a prefixed value between 0 and 1 to balance the positive labeled samples and negative labeled samples, and it is one of the most common ways to balance the classes. α_t takes values as shown in Eq. (3). Also, y ∈ {±1} specifies

the ground-truth class. The values of γ and α used are 2 and 1 respectively.

3.4 Training procedure

Since the learning rate is a very crucial hyperparameter, we used exponential decaying learning rates.

$$\alpha' = \alpha * e^{-k* \, epoch\, number} \tag{4}$$

where α' is the new learning rate while α is the initial learning rate and k is constant with the assigned value of 0.01.

In addition to this, we used another callback that included three model checkpoints to save the best model with the least validation loss, maximum validation macro F1 and micro F1. This ensured that we save the weights before the state when the model starts to overfit.

We tried to customize our model training by increasing the batch size in between the epochs. The models were trained in such a way that the batch size was doubled after every 15 epochs. An initial batch size of 32 was used which gradually reached 128 till the 30th epoch. The model was trained for 35 epochs. With smaller batch sizes, the gradients are a very rough approximation of the true gradients and so it takes a considerable amount of time to find the right solution. On the other hand, larger batches mean fewer parameter updates per epoch, so training is potentially much faster but they also tend to consume higher memory. Thus, considering the memory constraints of larger batch size, we didn't allow the batch size to exceed the limits beyond our storage resources but still furnishing us with the benefits. This training can also be taken as an ensemble of batch sizes that could find a trade-off between the memory availability and more accurate approximations of the true gradients. We also monitored how increasing the batch size at some point in training favored the metrics.

3.5 Classification on 5 classes

This subsection presents the methodology used to perform classification on 5 classes both before and after synthetic data augmentation.

3.5.1 Models and validation scheme used

For the subset of 5 classes, the training set consists of 47,606 images and the validation set had 11,902 images based on a fivefold cross-validation scheme. The CNN architectures that we have considered in the experiment included VGG16, ResNet50, NASNet Mobile and the InceptionV4. The VGG16 model outperformed other architectures on every metric. The training procedure used for the 5 classes is similar to the one mentioned earlier for 28 classes. A brief description of the classifier architecture used on top of the VGG16 model is as follows.

3.5.2 The best architecture on 5 classes

A single dense layer with 256 neurons and the ReLU activation function is used on the top of the feature extracting layers followed by a classifier with 5 neurons corresponding to the 5 target classes with sigmoid as an activation function. The entire model was then compiled using the focal loss and Adam optimizer with an initial learning rate of 0.0001. The metrics used were accuracy, macro F1 and micro F1 scores. The model then trained for 50 epochs yielded the best macro F1 score of 0.812 on the validation set.

4 Results and comparison

4.1 Results of image enhancement

Various image enhancement techniques were used as described in section A.1. To compare the performance of these techniques, 3 metrics were used: Mean Squared Error (MSE), the Peak-Signal-to-Noise-Ratio (PSNR) and the Structural Similarity Index (SSIM).

$$MSE(x,y) = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}\left(x_{ij} - y_{ij}\right)^2 \tag{5}$$

$$PSNR(x,y) = 10log_{10}\left(\frac{MAX^2}{MSE(x,y)}\right) \tag{6}$$

MSE is calculated using Eq. (5). Equation (6) shows that as MSE approaches zero, the PSNR value approaches infinity. This shows a higher quality image has a higher PSNR value. However, one of the critical limitations of PSNR is that it neglects the spatial relationship between the pixels and that it also ignores interpretation and difference in images which on the other hand is well perceived by the human visual system. The similarity between the two images is measured by SSIM developed by Wang et al. [40]. Equation (7) shows the formula for SSIM. It is recognized to be well correlated with the Human Visual System (HVS).

$$SSIM(x,y) = \frac{\left(2\mu_x\mu_y + c_1\right)\left(2\sigma_{xy} + c_2\right)}{\left(\mu_x^2 + \mu_y^2 + c_1\right)\left(\sigma_x^2 + \sigma_y^2 + c_2\right)} \tag{7}$$

4.2 Evaluation

The subsection presents the description of the evaluation metrics. The models were evaluated based on the following metrics - Macro F1, Micro F1 and Accuracy.

Pr(avg) = Average precision of all the classes, Re(avg) = Average recall of all the classes, TP = True positive, TN = True negative, FP = False positive, FN = False negative, P = Precision, R= Recall

4.2.1 Macro F1

$$MacroF1 = \frac{2 \times Pr(Avg) \times Re(Avg)}{Pr(Avg) + Re(Avg)} \tag{8}$$

4.2.2 Micro F1

$$P = \frac{\sum_c TP_c}{\sum_c TP_c + FP_c} \tag{9}$$

$$R = \frac{\sum_c TP_c}{\sum_c TP_c + FN_c} \tag{10}$$

Table 1 Results of various models on 28 classes

Models	Image size	Public LB	Private LB	Local CV
ResNet50	512X512	0.462	0.458	0.534
	256X256	0.448	0.437	0.523
ResNet101	512X512	0.475	0.459	0.538
	256X256	0.453	0.455	0.513
DenseNet121	512X512	0.490	0.472	0.535
	256X256	0.439	0.429	0.521
InceptionV3	512X512	0.540	0.520	0.591
	256X256	0.536	0.486	0.568
InceptionV4	**512X512**	**0.558**	**0.522**	**0.603**
	256X256	0.548	0.514	0.581
Xception	512X512	0.553	0.516	0.587
	256X256	0.551	0.502	0.572
NASNet Mobile	512X512	0.517	0.493	0.559
	256X256	0.516	0.468	0.539

Bold values represent the best values in each column of the tables

$$MicroF1 = \frac{2 \times P \times R}{P + R} \tag{11}$$

4.2.3 Accuracy

$$Accuracy = \frac{\sum_c TP_c + FN_c}{\sum_c TP_c + TN_c + FP_c + FN_c} \tag{12}$$

In Micro-average, the average metric is computed by taking contributions of all classes whereas in macro-average, independent metric evaluation for each class is done before taking the average (all classes treated equally). Hence, micro average accounts for class imbalance problems in a multi-class classification setup.

4.3 Results on the entire 28 classes

Table 1 shows the results of different classification models on all the 28 classes. Two different image sizes have been used: 256 pixels and 512 pixels. The metric used on Kaggle Public LB (Public Leaderboard), Kaggle Private LB (Private Leaderboard) and LocalCV (fivefold cross-validation score) is macroF1. The test data provided by Kaggle contained 11,702 images. The result on public LB is for around 29% of the test data while the private LB is calculated on 71% of the test data approximately. Table 1 depicts that the InceptionV4 model gave the best results on an image size of 512 pixels.

4.4 Results with classic data augmentation

In this subsection, we have summarized the results of various pre-trained classification models. Note that the results in this subsection are on the augmented images of the subset chosen using the classical augmentation techniques. The values in the table are the average of the respective metric obtained in each fold.

Table 2 shows that the VGG16 model gave the best performance on the metrics on the validation set. The set of pre-trained models used on 5 classes are different from those used for the classification task on 28 classes.

Table 2 Results of various models on different metrics using classical data augmentation

Metrics	Macro F1		Micro F1		Accuracy	
Models	Train	Test	Train	Test	Train	Test
VGG16	0.821	**0.812**	0.892	**0.871**	0.905	**0.884**
NASNet Mobile	**0.837**	0.778	**0.898**	0.826	**0.908**	0.820
Resnet50	0.805	0.794	0.878	0.845	0.871	0.853
InceptionV4	0.828	0.785	0.890	0.837	0.893	0.834

Bold values represent the best values in each column of the tables

The models trained on 28 classes have complex and deep architectures making them less suitable for training on 5 classes. These models have a craving for large amounts of data which otherwise could vulnerate them to overfitting. As it is evident from Table 2, architectures like NASNet Mobile and InceptionV4 gave inferior performance. This fact demotivated us to explore the rest of the models used for classifying 28 classes.

4.5 Results with synthetic data augmentation using GAN

In the following table, the results of the same models are summarized for synthetic augmentation using GAN. Table 3 holds the average values of the respective metrics on each fold. Figure 5 shows the generated images of DCGAN.

Table 3 Results of Various Models on Different Metrics Using Synthetic Data Augmentation

Metrics	Macro F1		Micro F1		Accuracy	
Models	Train	Test	Train	Test	Train	Test
VGG16	0.843	**0.834**	**0.914**	**0.894**	**0.924**	**0.906**
NASNet Mobile	**0.845**	0.791	0.907	0.847	0.913	0.843
Resnet50	0.825	0.813	0.898	0.862	0.89	0.876
InceptionV4	0.836	0.797	0.902	0.855	0.908	0.861

Bold values represent the best values in each column of the tables

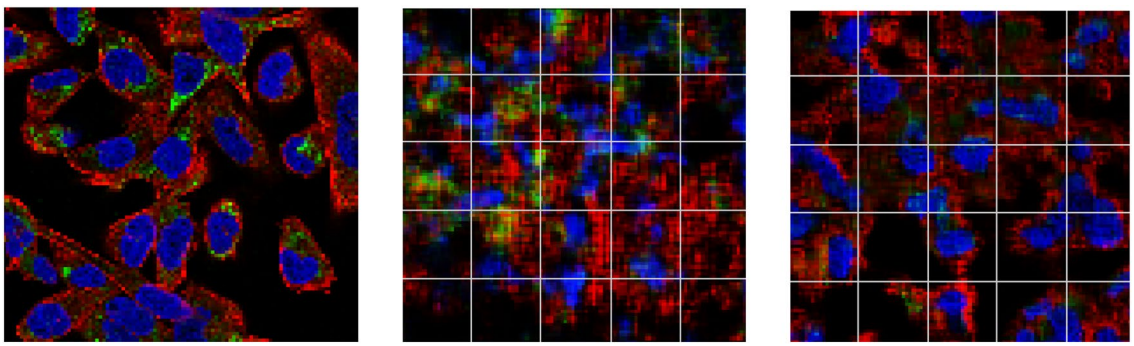

Fig. 5 Generated Images **a** Sample Input Image, **b** After 2000 epochs, **c** After 10,000 epochs

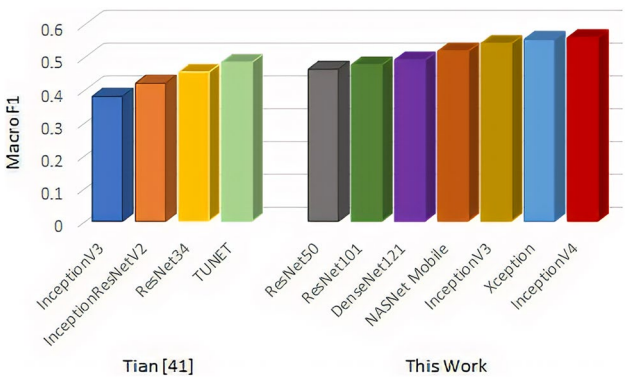

Fig. 6 Comparison with the existing work

4.6 Comparison with previous work

The subsection presents a comparison of our results with the existing research. Figure 6 shows the comparison of our work with the existing work. Twin U-Net (TUNet) model was used for the classification of the protein images [41]. A 9:1 split was used which accounts for 27,952 images in the train set and 3105 images in the validation set. In contrast to this, we have used the original competition data plus the external data thus accounting for 84,890 images in the train set and 21,222 images in each fold of the fivefold cross-validation. [41] achieved a macro F1 score of 0.483 on Kaggle privateLB whereas our best model InceptionV4 obtained the best macro F1 score of 0.522. Also, other models like ResNet34 and InceptionV3 gave inferior performance when compared to our pre trained models. In addition to this, no image enhancement techniques were used in [41].

5 Conclusions and future work

The work focused on applying various image enhancement techniques on microscopic images of protein and using these preprocessed images for the image classification task. The study also demonstrated the effectiveness of synthetic data augmentation using the generative model and compared it with the traditional augmentation methods.

We applied different image enhancement techniques like HE, AHE and CLAHE and compared them on different quality metrics. The CLAHE technique outperformed the other techniques with SSIM of 0.738. We used Deep Convolutional GAN (DCGAN) for synthetic data augmentation and compared the results with classical data augmentation. A significant improvement of 2.70% in macro F1 score and 2.64% in micro F1 score was witnessed for our best performing model VGG16. Other pre-trained models also showed similar improvements. Also, for 28 classes, the InceptionV4 model yielded the best macro F1 score of 0.603 on Local CV.

Even though we have achieved an appreciable score on the metrics used, there still exists room for improvement. There are some limitations to this work that could be addressed. In our work, there was an increase in the training complexity due to separate training of DCGAN for each protein class. In the future, we seek to apply this approach to other domains that could be benefitted from the availability of more synthetic data. In conclusion, we described an approach that automates and improves the classification of protein images. We believe the work would prove to be a boon in the domain of biotechnology.

Acknowledgements The authors are thankful to ABV-Indian Institute of Information Technology and Management, Gwalior for providing resources for this research work. This research work has not taken funding from any source.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Ethical Approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

1. Kraus OZ, Ba JL, Frey BJ. Classifying and segmenting microscopy images with deep multiple instance learning. Bioinformatics. 2016;32(12):i52–9.
2. LeCun Y, Bengio Y, Hinton G. Deep learning. Nature. 2015;521(7553):436–44.
3. Pärnamaa T, Parts L. Accurate classification of protein subcellular localization from high-throughput microscopy images using deep learning. Genes Genomes Genet. 2017;7(5):1385–92.
4. Tan J, Ung M, Cheng C, Greene CS. Unsupervised feature construction and knowledge extraction from genome-wide assays of breast cancer with denoising autoencoders. In: Pacific symposium on biocomputing co-chairs (2014). pp. 132–143.
5. Angermueller C, Pärnamaa T, Parts L, Stegle O. Deep learning for computational biology. Mol Syst Biol. 2016;12:7.
6. Sønderby SK, Sønderby CK, Nielsen H, Winther O. Convolutional LSTM networks for subcellular localization of proteins. In: International conference on algorithms for computational biology (2015 Aug 4), pp. 68–80. Springer, Cham.
7. Almagro Armenteros JJ, Sønderby CK, Sønderby SK, Nielsen H, Winther O. DeepLoc: prediction of protein subcellular localization using deep learning. Bioinformatics. 2017;33(21):3387–95.
8. Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition (2014 Sep 4). arXiv preprint arXiv:1409.1556.
9. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition 2016, pp. 770–778.
10. Szegedy C, Ioffe S, Vanhoucke V, Alemi AA. Inception-v4, inception-resnet and the impact of residual connections on learning. In: Thirty-first AAAI conference on artificial intelligence (2017 Feb 12).
11. Huang G, Liu Z, Van Der Maaten L, Weinberger KQ. Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition (2017), pp. 4700–4708.
12. Zoph B, Vasudevan V, Shlens J, Le QV. Learning transferable architectures for scalable image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition (2018), pp. 8697–8710.
13. Shin HC, Roth HR, Gao M, Lu L, Xu Z, Nogues I, Yao J, Mollura D, Summers RM. Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning. IEEE Trans Med Imaging. 2016;35(5):1285–98.
14. Tajbakhsh N, Shin JY, Gurudu SR, Hurst RT, Kendall CB, Gotway MB, Liang J. Convolutional neural networks for medical image analysis: Full training or fine tuning? IEEE Trans Med Imaging. 2016;35(5):1299–312.
15. Azizpour H, Sharif Razavian A, Sullivan J, Maki A, Carlsson S. From generic to specific deep representations for visual recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops (2015), pp. 36–45.
16. Penatti OA, Werneck RD, de Almeida WR, Stein BV, Pazinato DV, Júnior PR, Torres RD, Rocha A. Mid-level image representations for real-time heart view plane classification of echocardiograms. Comput Biol Med. 2015;1(66):66–81.
17. Sharif Razavian A, Azizpour H, Sullivan J, Carlsson S. CNN features off-the-shelf: an astounding baseline for recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops (2014), pp. 806–813.
18. Pizer SM, Amburn EP, Austin JD, Cromartie R, Geselowitz A, Greer T, ter Haar Romeny B, Zimmerman JB, Zuiderveld K. Adaptive histogram equalization and its variations. Comput Vis Graph Image Process. 1987;39(3):355–68.
19. Zuiderveld K. Contrast limited adaptive histogram equalization. In: Graphics gems IV (1994 Aug 1), pp. 474–485. Academic Press Professional, Inc.
20. Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems (2012), pp. 1097–1105.

21. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y. Generative adversarial nets. In: Advances in neural information processing systems (2014), pp. 2672–2680.
22. Radford A, Metz L, Chintala S. Unsupervised representation learning with deep convolutional generative adversarial networks (2015 Nov 19). arXiv preprint arXiv:1511.06434.
23. Denton EL, Chintala S, Fergus R. Deep generative image models using a laplacian pyramid of adversarial networks. InAdvances in neural information processing systems (2015), pp. 1486–1494.
24. Mirza M, Osindero S. Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784. 2014 Nov 6.
25. Odena A, Olah C, Shlens J. Conditional image synthesis with auxiliary classifier gans. In: Proceedings of the 34th international conference on machine learning-volume 70 (2017 Aug 6), pp. 2642–2651. JMLR. org.
26. Costa P, Galdran A, Meyer MI, Niemeijer M, Abràmoff M, Mendonça AM, Campilho A. End-to-end adversarial retinal image synthesis. IEEE Trans Med Imaging. 2017;37(3):781–91.
27. Dai W, Dong N, Wang Z, Liang X, Zhang H, Xing EP. Scan: Structure correcting adversarial network for organ segmentation in chest x-rays. In: Deep learning in medical image analysis and multimodal learning for clinical decision support (2018 Sep 20), pp. 263–273. Springer, Cham.
28. Xue Y, Xu T, Zhang H, Long LR, Huang X. Segan: adversarial network with multi-scale l 1 loss for medical image segmentation. Neuroinformatics. 2018;16(3–4):383–92.
29. Nie D, Trullo R, Lian J, Petitjean C, Ruan S, Wang Q, Shen D. Medical image synthesis with context-aware generative adversarial networks. In: International conference on medical image computing and computer-assisted intervention (2017 Sep 10), pp. 417–425. Springer, Cham.
30. Schlegl T, Seeböck P, Waldstein SM, Schmidt-Erfurth U, Langs G. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In: International conference on information processing in medical imaging (2017 Jun 25), pp. 146–157. Springer, Cham.
31. Alex V, KP MS, Chennamsetty SS, Krishnamurthi G. Generative adversarial networks for brain lesion detection. In: Medical imaging 2017: image processing (2017 Feb 24), Vol. 10133, p. 101330G. International Society for Optics and Photonics.
32. Ben-Cohen A, Klang E, Raskin SP, Amitai MM, Greenspan H. Virtual PET images from CT data using deep convolutional networks: initial results. In: International workshop on simulation and synthesis in medical imaging (2017 Sep 10), pp. 49–57. Springer, Cham.
33. Litjens G, Kooi T, Bejnordi BE, Setio AA, Ciompi F, Ghafoorian M, Van Der Laak JA, Van Ginneken B, Sánchez CI. A survey on deep learning in medical image analysis. Med Image Anal. 2017;1(42):60–88.
34. Greenspan H, Van Ginneken B, Summers RM. Guest editorial deep learning in medical imaging: overview and future promise of an exciting new technique. IEEE Trans Med Imaging. 2016;35(5):1153–9.
35. Shi J, Zhou S, Liu X, Zhang Q, Lu M, Wang T. Stacked deep polynomial network based representation learning for tumor classification with small ultrasound image dataset. Neurocomputing. 2016;19(194):87–94.
36. Salimans T, Goodfellow I, Zaremba W, Cheung V, Radford A, Chen X. Improved techniques for training gans. In: Advances in neural information processing systems (2016), pp. 2234–2242.
37. Kaggle. Human protein atlas image classification challenge. (2019). https://www.kaggle.com/c/human-protein-atlas-image-classification. Online Accessed on 24 Oct 2018.
38. Protein Atlas. The human protein atlas. https://www.proteinatlas.org. Online Accessed on 24 Oct 2018.
39. Lin TY, Goyal P, Girshick R, He K, Dollár P. Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision (2017), pp. 2980–2988.
40. Wang Z, Bovik AC, Sheikh HR, Simoncelli EP. Image quality assessment: from error visibility to structural similarity. IEEE Trans Image Process. 2004;13(4):600–12.
41. Tian Y. TUNet: Incorporating segmentation maps to improve classification (2019 Jan 27). arXiv preprint arXiv:1901.11379.