TELKOMNIKA Telecommunication, Computing, Electronics and Control
Vol. 18, No. 3, June 2020, pp. 1382~1388
ISSN: 1693-6930, accredited First Grade by Kemenristekdikti, Decree No: 21/E/KPT/2018
DOI: 10.12928/TELKOMNIKA.v18i3.14868

Transfer learning with multiple pre-trained network for fundus classification

Wahyudi Setiawan[1], Moh. Imam Utoyo[2], Riries Rulaningtyas[3]
[1]Informatics Department, University of Trunojoyo Madura, Indonesia
[2]Mathematics Department, Universitas Airlangga, Indonesia
[3]Physics Department, Universitas Airlangga, Indonesia

Article Info

Article history:

Received Sep 10, 2019
Revised Dec 14, 2019
Accepted Dec 22, 2019

Keywords:

Classification
Convolutional neural network
Multiple pre-trained network
Neovascularization
Transfer learning

ABSTRACT

Transfer learning (TL) is a technique of reuse and modify a pre-trained network. It reuses feature extraction layer at a pre-trained network. A target domain in TL obtains the features knowledge from the source domain. TL modified classification layer at a pre-trained network. The target domain can do new tasks according to a purpose. In this article, the target domain is fundus image classification includes normal and neovascularization. Data consist of 100 patches. The comparison of training and validation data was 70:30. The selection of training and validation data is done randomly. Steps of TL i.e load pre-trained networks, replace final layers, train the network, and assess network accuracy. First, the pre-trained network is a layer configuration of the convolutional neural network architecture. Pre-trained network used are AlexNet, VGG16, VGG19, ResNet50, ResNet101, GoogLeNet, Inception-V3, InceptionResNetV2, and squeezenet. Second, replace the final layer is to replace the last three layers. They are fully connected layer, softmax, and output layer. The layer is replaced with a fully connected layer that classifies according to number of classes. Furthermore, it's followed by a softmax and output layer that matches with the target domain. Third, we trained the network. Networks were trained to produce optimal accuracy. In this section, we use gradient descent algorithm optimization. Fourth, assess network accuracy. The experiment results show a testing accuracy between 80% and 100%.

Corresponding Author:

Riries Rulaningtyas,
Physics Department,
Universitas Airlangga, Surabaya, Indonesia.
Email: riries-r@fst.unair.ac.id

1. INTRODUCTION

A system requires learning process to perform certain tasks. The tasks include image enhancement, classification, clustering, recognition, and detection. Data processing needs to do it. Data is divided into two parts, training and testing data. In conventional systems, training data processed to get knowledge. The problem arises when the amount of training data is limited, the learning process doesn't well perform. An alternative solution to the problem is transfer learning. It is a machine learning method that works by utilizing existing models. Transfer learning modifies and updates parameters on the model. Transfer learning makes modified models as learning with different tasks. The model used for transfer learning has learned from other data, so learning is not needed from scratch. The model has recognized features such as textures, shapes, and colors as a result of previous learning.

The benefit of transfer learning is well learning even though limited training data. Contrast to traditional machine learning, every learning process always requires relatively large amounts of data [1]. The difference between traditional machine learning and transfer learning is found in Figure 1. Nowadays, transfer learning has been applied to robotics [2, 3] image classification [4, 5], sentiment classification [6], game technology [7, 8] and text classification [9]. Generally, the type of transfer learning used in deep learning is a pre-trained network. The phase for conducting transfer learning as follows:

- Select a specific model. Pre-trained network models are taken from existing models.
- Reused model. Pre-trained models can be used as a starting point for carrying out a new task. A new task can use the whole part of a pre-trained model or partly depends on system requirements.

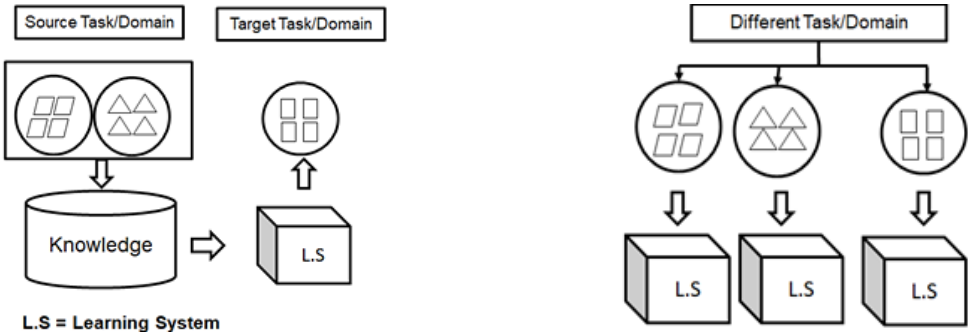

Figure 1. (a) Traditional machine learning, (b) Transfer learning, modified from

- Modification of the model. Modifications are made at the last fully connected layer

This paper discussed classification of fundus images. Classification for distinguishing normal and neovascularization. Neovascularization is the appearance of new vessels in optic disk or other surfaces of retina. Neovascularization features of the blood vessels are brittle, irregular in shape, and easily lost. Neovascularization is a severe diabetic retinopathy (DR). Neovascularization consists of two categories: neovascularization on the disc (NVD) and neovascularization elsewhere (NVE). NVD is a new vessel in the optic disc while NVE is a new vessel in the entire surface of the retina except in optic disc [10]. neovascularization of the fundus images is shown in Figure 2.

There are previous studies that classified fundus images. Tennakoon et al. classified two categories: gradable and ungradable based on image quality fundus. The model uses shallowNet and a modified AlexNet model. A fully connected layer (FCL) fc7 is a layer for feature extraction. FCL fc8 is the fine-tuning layer for classification. Classification using SVM, boosted tree, and k-NN methods. The data consists of 463 images. The highest accuracy is 98.27% using shallowNet [11]. Li et al. classified fundus images using data from DR1 and MESSIDOR. The amount of data for each dataset is 1,014 and 1,200 images. There are three steps for transfer learning, fine-tuning all layers on pre-trained CNN models according to their functions, fine-tuning pre-trained CNN models on additional layers, then feature extraction and classification using SVM.

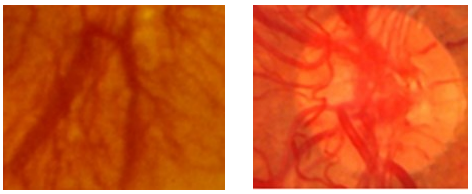

Figure 2. Neovascularization in fundus patch; (a) NVE, (b) NVD [12]

The experiment uses several models, AlexNet, googleNet, and VGG. The parameters used were maximum epoch 30, minibatch size 50, learning rates 0.1 to 0.0001, weights 0.0005 and momentum 0.95. Optimization using stochastic gradient descent with momentum (SGDM) algorithm. The test results showed the best accuracy is the modified VGG-m model of 95.49% for the DR1 dataset and GoogleNet modification of 79.37% for the MESSIDOR dataset [13].

Choi et al. [14] classified 10 classes of diabetic retinopathy (DR). Data consists of 10,000 images. Each category has 1,000 images. The model used for transfer learning is VGG19 and AlexNet. The optimization algorithm uses SGDM, momentum 0.9 learning rate 10-6, and max epoch 50. The test scenario is varied, with 3 and 5 classes. The test results showed the best accuracy is VGG19 for classification of three categories at 80.8%, while classification of 5 categories showed the highest accuracy of 59.1% [14]. Masood et al. classified 4 DR classed as mild, moderate, severe non-proliferative diabetic retinopathy (NPDR), and PDR. The dataset is taken from eyePacs. The steps for training learning are preprocessing and retraining Inception-V3. Result shows 48.2% for accuracy [15].

Oktalora et al. [16] classify for exudate. Exudate is a symptom in the form of a yellow spot, irregular shape, arising from lipid infiltration in the retina. Exudate is a symptom of diabetic retinopathy. This study uses a LeNet model with seven layers. Experiment data using Optha dataset. The size of data is 48x48 pixels. The classification consists of two categories: normal and exudate [16]. Sadek et al. build transfer learning to classify 3 categories include normal, exudates, and drusen. The dataset uses are STARE, HRF, DrisonDB, Optha, HEIMED and MESSIDOR dataset. Transfer learning uses modified VGG, GoogleNet and ResNet models. Result shows average accuracy from 91.23% to 92% [17].

The above studies have not reached the optimal accuracy. Characteristics of diabetic retinopathy (DR) disease have not been fully classified. The characteristics of DR are microaneurysm, hemorrhages, exudates, cotton wool spots, and neovascularization. The novelty of this study is classification of fundus images to distinguish normal and neovascularization using transfer learning. Besides, novelty is also found in CNN modification technique by utilizing the last three layers of each model. The results of measurement accuracy from transfer learning are compared in the optimization of gradient descent such as stochastic gradient descent with momentum (SGDM), root mean square propagation (RMSProp), and adaptive moment optimization (Adam).

2. RESEARCH METHOD

The experiment data consists of 2 classes include normal and neovascularization. Each class has 50 patches, so the total data is 100 patches. It is taken from the MESSIDOR [18] and retina image bank [12]. The pre-trained network is a CNN model. CNN is the same as the other neural networks, consisting of weight, bias and activation functions. CNN has 2 big parts of the layer, layer for feature extraction and layer for classification. The layer for feature extraction consists of a convolutional layer, pooling layer, stride, and padding. While layer for classification consists of fully connected layer, softmax, and output layer [19]. Pre-trained network becomes a part of transfer learning. Phase of transfer learning is import pre-trained network, replace classification layer, train network on the fundus image data, and get performance measure of accuracy.

2.1. Import pre-trained network

We use AlexNet [19], VGG16, VGG19 [20], ResNet50, ResNet101 [21], GoogleNet [22], Inception-V3 [23], Inception_ResNetV2 [24], and Squeezenet [25] as pre-trained network. The pre-trained network has trained in ImageNet competition. It has more than a million images and 20,000 classes [26]. Each pre-trained network has a different layer configuration. The layer at the beginning and middle called a feature extraction layer. These layers produce simple features such as brightness and edges, to complex unique features such as colors and shapes. The results of feature extraction layer at source domain can be transferred for feature extraction layer at target domain. Feature extraction learning on target domains also knows training fundus data images. Figure 3 shows a proposed framework of transfer learning for fundus image classification.

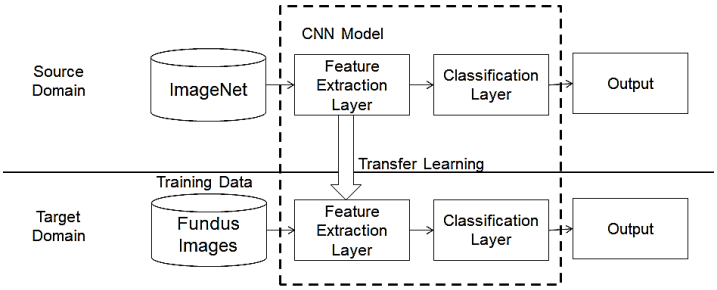

Figure 3. The proposed framework of transfer learning for fundus image classification

2.2. Replace classification layer

The classification layer is known as 3 final layers i.e fully connected layer, softmax, and an output layer. It replaced by a pre-trained network and substitutes with new classification layer that matched with a new classification task. It includes new number of classes and a set learning rate in the new network. There are exceptions for squeezenet, a layer that must be replaced consists of five layers. Table 1 shows the classification layer that replaces the network.

Table 1. Classification layer of the pre-trained network

Pre-Trained Network	Classification Layer
AlexNet	fc8, prob, output
VGG16	fc8, prob, output
VGG19	fc8, prob, output
ResNet50	fc1000, fc1000_softmax, classificationlayer_fc1000
ResNet101	fc1000, prob, classificationlayer_predictions
GoogleNet	loss3-classifier, prob, output
Inception-V3	predictions, predictions_softmax, classificationlayer_predictions
Inception-ResNetV2	predictions, predictions_softmax, classificationlayer_predictions
Squeezenet	conv10, relu_conv10, pool10, prob, classificationlayer_predictions

2.3. Train network on fundus image

Data is processed with research method as shown in Figure 3. It is a phase of transfer learning with a pre-trained network for classification of fundus images. At the top, source domains are pre-trained networks that have classified data on ImageNet. The train network also needs an optimization algorithm. We use an optimization gradient descent algorithm. Gradient descent (GD) obtain optimal parameter weights, reduce prediction errors and improve predictions of accuracy. GD performs parameter optimization on the network. Besides, GD has a linear complexity of data increment. GD can be computed in parallel by utilizing a graphical processing unit (GPU). The application of GD on the CNN model proves that GD can do training with millions of data [27].

2.4. Gradient descent with momentum

Momentum is a method for GD acceleration by utilizing gradient information in the previous steps. Accumulation of gradients is useful for controlling oscillatory effects**.** Furthermore, it is expected that the optimization path can be more stable [28].

Algorithm 1. (Gradient descent with momentum)

1. $m_0 = 0$
2. $g_t := \nabla_{\theta_{t-1}} L(\theta_{t-1})$
3. $m_t := g_t + \beta m_{t-1}$
4. $\theta_t := \theta_{t-1} - \alpha\, m_t$

with g_t = gradient loos function to θ_{t-1}, θ_t = next parameter, α = learning rate.

The constant controls the size of the contribution from the previous gradient. Generally, set to 0.9 is the best value of the experiment that has been carried out. If set to 0, then the GDM results are the same as GD. Stochastic gradient descent with momentum (SGDM) is a variant of GDM. The difference is data access. If in GDM the data is processed all the data at the same time. Data on SGDM will be processed suitably with minibatch size [27].

2.4.1. AdaGrad and RMSProp

Adaptive subgradient descent (AdaGrad) [29] get GD improvements by providing different update speeds for each vector dimension. The AdaGrad algorithm is found in algorithm 2 [28]. Constants provide information about changing the value of an element in the gradient vector. If the value in a certain dimension decreases, the update speed in certain dimensions increases and vice versa. This will balance the contribution of each dimension of the gradient vector so that the optimization path becomes more stable

Algorithm 2. (AdaGrad)

1. $n_0 = 0$
2. $g_t := \nabla_{\theta_{t-1}} L(\theta_{t-1})$
3. $n_t := g_t^{\,2} + n_{t-1}$
4. $\theta_t := \theta_{t-1} - \alpha \frac{g_t}{\sqrt{n_t + \varepsilon}}$

with n_t = *adaptive subgradient*, ε =a constant 1e-6

The problem with AdaGrad is the value can be very large at certain time. It will slow down the optimization process time. The solution to these problems is to modify by adding constants. These

constants are used to set variable quantities. The root means square propagation (RMSProp) algorithm is found in algorithm 3 [28, 30].

Algorithm 3. (RMSProp)

1. $n_0 = 0$
2. $g_t := \nabla_{\theta_{t-1}} L(\theta_{t-1})$
3. $n_t := (1-\gamma){g_t}^2 + \gamma n_{t-1}$
4. $\theta_t := \theta_{t-1} - \alpha \frac{g_t}{\sqrt{n_t}+\varepsilon}$

with γ = *koefisien decay rate* 0,95

2.4.2. Adaptive moment optimization (Adam)

Adam algorithm combines the two approaches to improve GD, momentum and adaptive subgradient. This algorithm combines GDM with RMSProp. Adam's algorithm is shown in algorithm 4 [28]. Line 4 is an element of momentum, line 6 is an adaptive subgradient element. Adam has a correction bias technique with a better approximation [31].

Algorithm 4. Adam

1. $n_0 = 0$
2. $g_t := \nabla_{\theta_{t-1}} L(\theta_{t-1})$
3. $m_t := (1-\beta)g_t + \beta m_{t-1}$
4. $\widehat{m}_t := \frac{m_t}{1-\beta^t}$
5. $n_t := (1-\gamma){g_t}^2 + \gamma n_{t-1}$
6. $\hat{n}_t := \frac{n_t}{1-\gamma^t}$
7. $\theta_t := \theta_{t-1} - \alpha \frac{\widehat{m}_t}{\sqrt{\hat{n}_t}+\varepsilon}$

m_t = momentum, n_t = adaptive subgradient, $\widehat{m}_t$ = momentum estimation with *corrected bias* at time *t,* $\hat{n}_t$ = = adaptive subgradient estimation with corrected bias at time *t.*

3. RESULT AND DISCUSSION

The experiment includes the following scenarios:

- Data divide two parts, 70% for training and 30% for testing. Total data is 100 patches, 70 patches for training and 30 patches for testing
- Training phase. Make sure the image size at the training and validation suitable with a pre-trained model. If it is not yet sized, then resize the image. Another alternative is to get an augmentation image to automatically suitable with the image input size.
- The training parameters are set as follows: learning rate 1e-4, minibatch size 4, max epoch 5, validation frequency 3.

The results of an experiment are shown in Tables 2, 3, and 4. Table 3 shows validation using SGDM algorithm produces the best accuracy up to 100% using VGG16 with a time of 16,572 seconds. In Table 4, validation with RMSProp produces the best accuracy value of up to 93.3% with a time of 164.38 seconds. Pre-trained Network used is Resnet50. Table 5 shows that validation using Adam algorithm produces best accuracy of 96.7%. The experiment using Alexnet with a processing time of 36,274 seconds.

The initialization of learning rate, minibatch size, max epoch, validation frequency, and gradient descent optimization algorithm are factors that influence results of validation accuracy and processing time. Learning rate can be initialized starting from big value until it gradually shrinks. Learning rate is between 0 and 1. If learning rate too large, it will cause overfitting, while learning rate value is too small which will cause a longer processing time.

Table 2. Accuracy and time processing with SGDM

Pre-trained Network	Accuracy (%)	Time (Second)
Alex Net	93.3	8.9537
googLeNet	86.7	12.876
ResNet50	100	49.411
VGG16	100	16.572
VGG19	80	18.163
ResNet101	93.3	126.61
Inception-V3	96.7	95.715
InceptionResNetV2	70	352.05
Squeezenet	96.7	4.0363

Table 3. Accuracy and time processing with RMSProp

Pre-trained Network	Accuracy (%)	Time (Second)
AlexNet	83.3	26.957
googLeNet	90	73.062
ResNet50	93.3	164.38
VGG16	53.3	246.06
VGG19	50	225.00
ResNet101	93.3	305.7
Inception-V3	90	230
InceptionResNetV2	90	530.06
Squeezenet	53.3	18.896

Table 4. Accuracy and time processing with adam

Pre-trained Network	Accuracy (%)	Time (Second)
AlexNet	96.7	36.274
googLeNet	93.3	64.961
ResNet50	90	149.93
VGG16	50	126.31
VGG19	50	388.24
ResNet101	86.7	196.91
Inception-V3	86.7	252.1
InceptionResNetV2	93.3	448.99
Squeezenet	96.7	44.925

Table 5. Result comparison with the previous study

Author	Class	Pre-trained Network	Accuracy (%)
Tennakoon et al. [11]	2	AlexNet	98.27
Li et al. [13]	2	VGGm, GoogleNet	95.49 79.39
Choi et al. [14]	3	VGG19	80.8
	5	VGG19	59.1
Masood et al. [15]	4	Inception V3	48.2
Sadek et al. [17]	3	VGG, GoogleNet, ResNet	91.23-92
Proposed Method	2	AlexNet	96.7
		GoogLeNet	93.3
		ResNet50	100
		VGG16	100
		VGG19	80
		ResNet101	93.3
		Inception-V3	96.7
		InceptionResNetV2	93.3
		Squeezenet	96.7

Minibatch size will affect memory usage during processing. Smaller minibatch size requires less memory when processing. Generally, minibatch size is 2^n. Max epoch value is maximum value that can be done to process one feedforward on CNN. Iteration stops when an error is constant or when max epoch is reached. Validation frequency is value given for the number of validation frequency. These values can be varied to obtain optimal accuracy and minimal processing time. Table 5 shows comparison between the methods in this article with previous studies. In this article, transfer learning was done with multiple pre-trained networks include multiple pre-trained networks. The results showed validation of up to 100% accuracy using ResNet50 and VGG16.

4. CONCLUSION

Transfer learning using multiple pre-trained networks has been made to determine the category of fundus images including normal and neovascularization. It is used 100 patches taken from MESSIDOR and Retina Image Bank. Transfer learning can be used as an option to increase validation accuracy. The experiment result shows the best classification is found in transfer learning using pre-trained network VGG16 with validation accuracy up to 100% and time processing 16,572 seconds. For further research, we can use own CNN model. The amount of data and number of classes need to be enlarged for validation reliability of CNN model.

REFERENCES

[1] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineer*ing, vol. 22, no. 10, pp. 1345-1359, October 2010.
[2] M. K. Helwa and A. P. Schoellig, "Multi-Robot Transfer Learning : A Dynamical System Perspective," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4702-4708, 2017.
[3] B. Botond and J. Peters, "Alignment-based Transfer Learning for Robot Models," *The 2013 International Joint Conference Neural Networks (IJCNN),* 2013.
[4] Y. Zhu, Y. Chen, and Z. Lu, "Heterogeneous Transfer Learning for Image Classification," *Twenty-Fifth AAAI Conference on Artificial Intelligence Heterogeneous*, pp. 1304-1309, 2008.
[5] B. Petrovska, I. Stojanovic, and T. Atanasova-pacemska, "Classification of Small Sets of Images with Pre-trained Neural Networks," *Int. J. Eng. Manuf.*, vol. 4, pp. 40-55, 2018.

[6] Y. Yoshida, T. Hirao, T. Iwata, M. Nagata, and Y. Matsumoto, "Transfer Learning for Multiple-Domain Sentiment Analysis-Identifying Domain Dependent/Independent Word Polarity," *AAAI Conference on Artificial Intelligence*, pp. 1286-1291, 2011.

[7] M. Sharma, M. Holmes, J. Santamaria, A. Irani, C. Isbell, and A. Ram, "Transfer Learning in Real-Time Strategy Games Using Hybrid CBR/RL," *IJCAI*, pp. 1041-1046, 2005.

[8] T. R. Hinrichs and K. D. Forbus, "Transfer Learning through Analogy in Games," *Ai Magazine*, vol. 32, no. 1, pp. 70-83, 2011.

[9] C. B. Do and A. Y. Ng, "Transfer learning for text classification," *Conference: Advances in Neural Information Processing Systems 18 [Neural Information Processing Systems]*, 2005.

[10] W. Setiawan, M. Utoyo, and R. Rulaningtyas, "Classification of neovascularization using convolutional neural network model," *TELKOMNIKA Telecommunication Computing Electronics and Control*, vol. 17, no. 1, pp. 463-473, 2019.

[11] R. Tennakoon and P. Roy, "Image Quality Classification for DR Screening Using Convolutional Neural Networks," *Proceedings of the Ophthalmic Medical Image Analysis International Workshop*, pp. 113-120, 2016.

[12] American Society of Retina Specialists, "Retina Image Bank," [Online]. Available: https://imagebank.asrs.org

[13] X. Li, T. Pang, B. Xiong, W. Liu, P. Liang, and T. Wang, "Convolutional Neural Networks Based Transfer Learning for Diabetic Retinopathy Fundus Image Classification," *10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, no. 978, 2017.

[14] J. Y. Choi, T. K. Yoo, J. G. Seo, J. Kwak, T. T. Um, and T. H. Rim, "Multi-categorical deep learning neural network to classify retinal images : A pilot study employing small database," *PLoS One*, pp. 1-16, 2017.

[15] S. Masood and T. Luthra, "Identification of Diabetic Retinopathy in Eye Images Using Transfer Learning," *International Conference on Computing, Communication and Automation (ICCCA2017*), no. 2, pp. 1183-1187, 2017.

[16] S. Oktalora, O. Perdomo, F. Gonzales, and H. Muller, "Training Deep Convolutional Neural Networks with Active Learning for Exudate Classification in Eye Fundus Images," *CVII-STENT/LABELS 2017, LNCS 10552*, pp. 146-154, 2017.

[17] I. Sadek, M. Elawady, A. El, and R. Shabayek, "Automatic Classification of Bright Retinal Lesions via Deep Network Features," *ArXiv*, pp. 1-20, 2017.

[18] E. Decencière et al., "Feedback on a Publicly Distributed Image Database: the Messidor Database," *Image Anal. Stereol.,* vol. 33, no. 3, pp. 231, 2014.

[19] A. Krizhevsky and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Advances in neural information processing systems*, vol. 25, no. 2, pp. 1-9, 2012.

[20] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *ICLR*, pp. 1-14, 2015.

[21] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 770-778, Las Vegas, NV, 2016.

[22] C. Szegedy et al., "Going deeper with convolutions," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1-9, 2015.

[23] Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* Las Vegas, NV, pp. 2818-2826, 2016.

[24] C. Szegedy, S. Ioffe, and V. Vanhoucke, "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning," *ArXiv*, pp. 1-12, 2016.

[25] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size," *ICLR,* pp. 1-13, 2017.

[26] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and F.-F. Li, "ImageNet: A large-scale hierarchical image database," *IEEE Conf. Comput. Vis. pattern Recognit*, pp. 248-255, 2009.

[27] S. Ruder, "An overview of gradient descent optimization," *ArXiv,* pp. 1-14, 2017.

[28] M. Ghifary, "Deep Learning Optimization," 2017. [Online]. Available: https://ghif.github.io/aiml/2017/04/11/optimisasi-pada-deep-learning.html. [Accessed: 20-Aug-2018].

[29] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *J. Mach. Learn. Res*., vol. 12, pp. 2121-2159, 2011.

[30] G. E. Hinton, N. Srivastava, and K. Swersky, "Lecture 6a- overview of mini-batch gradient descent," *COURSERA Neural Networks Mach. Learn.*, pp. 31, 2012.

[31] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *ICLR*, pp. 1-15, 2015.