An Enhanced Slim-CNN for Classifying Floods and Garbage Images in the Jakarta Super-App Platform

Irfan Dwiki Bhaswara*, Yudhistira Nugraha*†, Andy Ernesto*, Farizah Rizka Rahmaniar*,
Andi Sulasikin*, Juan Intan Kanggrawan*, Alex L. Suherman*‡
*Jakarta Smart City, Department of Communications, Informatics, and Statistics, Jakarta, Indonesia.
†School of Computing, Telkom University, Bandung, Indonesia.
‡Directorate of Research and Community Services, Telkom University, Bandung, Indonesia.

***Abstract*—Jakarta, as the capital city of Indonesia, has facing the same problems every year regarding the issues of floods and garbage. Many innovative solutions have been made until now incl. One of the solutions is creating JAKI as the Jakarta Super-App platform, which can support public complaints based on geotagging photo. However, it still has lacked that the pictures uploaded by the user are incorrect with the categories. This paper attempts to solve by creating a model that automatically classifies the categories based on the image, especially for flood and garbage image. We propose an approach that uses a deep learning model called SlimCNN. We perform some enhancements on the SlimCNN model called Enhanced SlimCNN. In addition, we visualise our model with class activation maps. The result shows that our enhancement model achieves an accuracy of 96.7% on the JAKI database image. The findings of this study perform better results than the regular SlimCNN and other models, such as Alexnet and Squeezenet. Thus, Enhanced SlimCNN demonstrates promising results as a model solution for the development of Jakarta Super-App.**

***Index Terms*—Deep learning; Image classification; Class activation maps, SlimCNN**

I. Introduction

Data is a valuable asset today. With abundant data, current technologies can open up the possibility of making predictions based on the information obtained into something valuable and useful to several parties. Data can also be a tool to help people in the field of health, disasters, and community reports to solve many problems in a city, and it can help almost all aspects. Moreover, according to McKinsey Global Institute, data can create values that support humans in making a decision using sophisticated algorithms [1].

Many cities have already used their data to understand urban problems for many decades [2]. An example of this can be seen in the city of Jakarta. The Jakarta Regional Government, from year to year, continues to strive to solve problems in their area. One of the problems is flood and garbage. This problem is the three most extensive public reports throughout 2020 received by Jakarta Regional Government Officials as depicted in Figure 1. To solve this problem, the people of Jakarta use Jakarta Kini (JAKI) application as a reporting tool for flood and garbage problems in Jakarta. The JAKI application uses geotagging photo so people can report based on the categories and the location itself.

However, the geotagging photo method still has some limitation. One of the downsides is improper pictures that do not fit the categories presented, such as a person selfie image tagged as a flood image. The Jakarta government tries to implement a technology that can recognise the categories of these objects automatically. This automatic labelling for each category will be significant since some people still do not understand the report's category. In addition, it will also eliminate improper pictures. Therefore, this technology will be beneficial for officers who will follow up on the report.

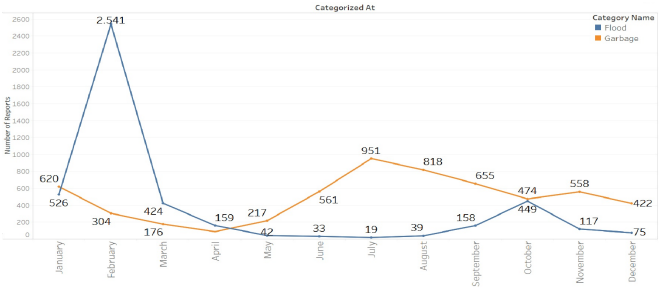

Fig. 1. The number of complaints related to Flood and Garbage in Jakarta throughout 2020.

We chose Jakarta as our study since Jakarta is already proven to handle citizen reports by using the JAKI application. However, some JAKI reports look unclear, so the reporting system admin sometimes has to read more on the photo reports. Therefore, we apply image processing to improve the report's quality and make it easier for all parties without reducing report quality.

In this paper, we present two contributions to handle this problem. First, we use the SlimCNN model [3] with some enhancements on the first layer to classify flood and garbage images. Second, we also apply class activation maps (CAMs) [4] to visualise which region of convolutional neural network (CNN) is activated according to its classes. In other words, the CAMs technique allows us to see which location of the object instantly from CNN.

The rest of the paper is organised as follows. Section II describes the related work in this paper. Section III outlines the methodology that is used in this research. In Section IV, some experiment setups are explained as well as the results. Finally, Section V provides the conclusions of the research work.

II. Related Work

Image classification has been categorised as a fundamental problem in computer vision. The accuracy of classifying an object is the primary concern. Previously, local hand-crafted feature extraction combined with a machine learning algorithm is used as a standard image classification technique [5] [6]. The robustness descriptor of the local binary pattern (LBP) [7], as an example, is often used for feature extraction with a combination of support vector machine (SVM) [8] as learning features. However, the local hand-crafted method brings some disadvantages. One of the disadvantages is that LBP sometimes misses the local structure as LBP only considers the neighbourhood pixel [9].

In recent years, deep learning has tremendously solved many problems, especially in image classification research task. For example, in the last few years, at the beginning era of deep learning, the Alexnet [10] model reaches the top-1 accuracy of 63.3% on the ImageNet dataset [11]. Another example comes from a recent model called Meta Pseudo Labels [12] that achieves top-1 accuracy of 90.2% on the ImageNet dataset.

Of many successful models, deep learning for image classification stands on the giant's shoulders of CNN model. Starting on LeNet-5 [13], Alexnet and VGG16 [14] use a conventional CNN model, where one convolutional layer is stacked with another layer, so it resembles a pyramid-like shape. Although there is a new method on image classification using a transformer similar to in natural language processing task [15], the CNN model is still be used today since it is very effective and efficient in extracting features of an image [16].

However, a recent study shows that in conventional CNN, which is only stacking convolution layer with another convolution layer, it will not increase the accuracy. It happens since a vanishing gradient occurs on a deeper network [17]. In order to overcome such a problem, a ResNet model [18] introduces a new method called skip connections. The skip connections provide a convolution layer to skip one or two convolution layer next to it.

At this time, deep learning for image classification is started to moving into mobile phone domains. Many architectures are designed to have low memory storage but have high accuracy, such as Squeezenet [19]. The Squeezenet has accuracy equivalent to Alexnet, but it has only 50× fewer parameters than Alexnet. Furthermore, SlimCNN [3] can reduce more parameters with higher accuracy than Squeezenet, where it uses skip connections as its backbone model.

Although many image classifications based on deep learning have been developed in many works of literature, it is only tested on common objects, such as car, person, cat, and dog. None of them is tested in the specific object, for example, detecting flood and garbage image. Therefore, we attempt to solve this problem by applying SlimCNN in such a specific scenario.

TABLE I
Enhanced SlimCNN Architecture

Layer	Input	Output	Kernel	Stride	Padding
Conv + BN	3	64	3	1	1
Max Pooling	-	-	2	2	-
Slim Module	64	16	1,3[a]	1	0,1[b]
Max Pooling	-	-	2	2	-
Slim Module	48	32	1,3[a]	1	0,1[b]
Max Pooling	-	-	2	2	-
Slim Module	96	48	1,3[a]	1	0,1[b]
Max Pooling	-	-	2	2	-
Slim Module	144	64	1,3[a]	1	0,1[b]
Max Pooling	-	-	2	2	-
Slim Module	192	80	1,3[a]	1	0,1[b]
Conv + BN	240	240	3	1	1
Conv	240	240	3	1	1
Global Average Pooling					
Fully Connected	240	2	-	-	-

Notes:
[a] Use kernel size = 1 for squeeze and expand layer, use kernel size = 3 for depth-wise layer.
[b] Use padding = 0 for squeeze and expand layer, use padding = 1 for depth-wise layer.
[c] All layers use ReLU activation function.

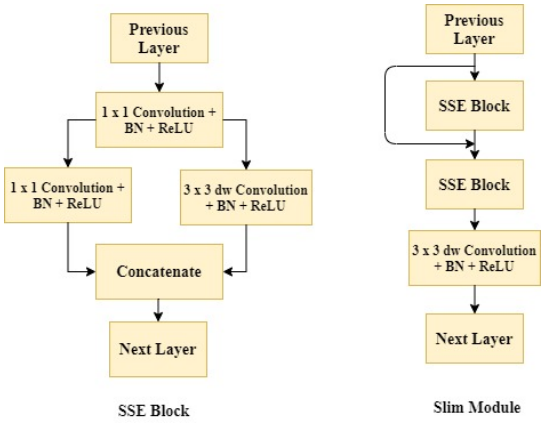

Fig. 2. SSE block and Slim Module.

III. Methodology

In this section, we show the detail of Enhanced SlimCNN architecture and an explanation behind CAMs. We also explain a mathematical detail for the CAMs method.

A. Enhanced SlimCNN

Enhanced SlimCNN is an enhancement from the SlimCNN model. The architecture follows the regular SlimCNN. This enhancement model consists of one convolutional layer with Batch Normalization (BN) [20] and rectified linear unit (ReLU) [21] and 4 Slim module layers as shown in Table I.

TABLE II
RAW DATASET

Category	Image URL
Garbage	https://s3-jaki.jakarta.go.id/jaki/report/media/8926f8f2-8fe0-423c-b427-3e8373bcdf1c
Garbage	https://s3-jaki.jakarta.go.id/jaki/report/media/e39ff178-6311-4fb5-80d8-80a4b9e25ee0
⋮	⋮
Flood	https://s3-jaki.jakarta.go.id/jaki/report/media/391b2e10-e859-4ac8-9add-03fa1fd084fa
Flood	https://s3-jaki.jakarta.go.id/jaki/report/media/bb017cdf-1aac-4f99-acc0-d29a1de77e1b

Fig. 3. Some image dataset obtained from the JAKI database.

Inside the Slim module, there are two separable squeeze expand (SSE) blocks with skip connections and one 3×3 depth-wise (dw) separable convolution [22]. This SSE block is similar to the Fire module in Squeezenet. However, instead of using regular 3×3 convolutions in the expand layer, it replaces 3×3 depth-wise separable convolution. Figure 2 shows the SSE block and Slim module, respectively.

To train the Enhanced SlimCNN, we use the binary cross-entropy loss function. This function is shown in Equation 1 where N is the number of classes, y_i is the label of the classes, and $p(y_i)$ is the predicted probability of the classes.

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N} y_i \log p(y_i) + (1 - y_i) \log(1 - p(y_i)) \quad (1)$$

$$GAP = \frac{1}{K}\sum_{i=1}^{u}\sum_{j=1}^{v} F_{i,j}^{N} \quad (2)$$

$$CAM = \frac{1}{K}\sum_{z=1}^{l}\sum_{i=1}^{u}\sum_{j=1}^{v} w_z^c F_{i,j}^{N} \quad (3)$$

TABLE III
NUMBER OF PARAMETERS FOR ALL MODELS

Model	Number of Parameters (M)	Storage Size (MB)	Testing Accuracy
SlimCNN	0.564	2.3	0.9199
Enhanced SlimCNN	2.04	8.00	0.9670
Squeezenet	0.723	2.77	0.9274
Alexnet	2.47	9.42	0.8719

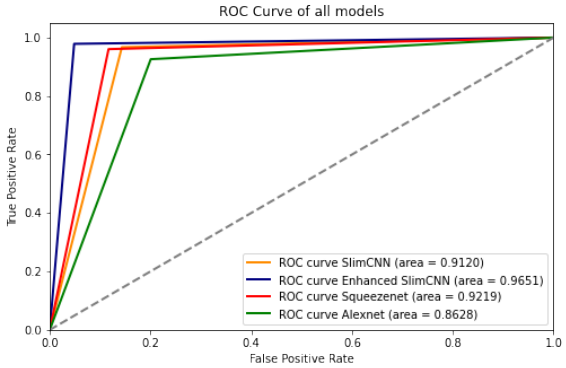

Fig. 4. ROC curve result for all models.

B. Class Activation Maps

Class activation maps help understand which part of the image regions used by CNN to identify a relevant class in the image. These image regions are activated in pixel-by-pixel and represented by a heatmap. The CAMs technique also benefits in doing object localisation without using any labelling or bounding box coordinate since the CNN directly shows which class location is presented in the image.

Some modifications are needed to get the CAMs on the last layer of CNN. Typically, the last layer of CNN is connected directly to a fully connected (FC) layer. However, in CAMs, we add Global Average Pooling (GAP) after the last convolutional layer before the FC layer. This GAP will calculate the average of feature maps from the last of the convolutional layer [23]. The mathematical notation of GAP is shown in Equation 2, where K is a total number of features, F^N is the feature maps at N-layer, and u and v are the width and height of the feature maps.

Later, if the GAP equation is multiplied together with the pretrained weight w_z^c that is corresponding to its classes, the Equation 2 can be expanded to get class activation maps as depicted in Equation 3 [4].

IV. RESULTS AND DISCUSSION

All training and testing were done using Pytorch [24] and OpenCV [25]. The experiments measured and compared Enhanced SlimCNN performances to other models, for example regular SlimCNN, Squeezenet, and Alexnet, for classifying flood and garbage images.

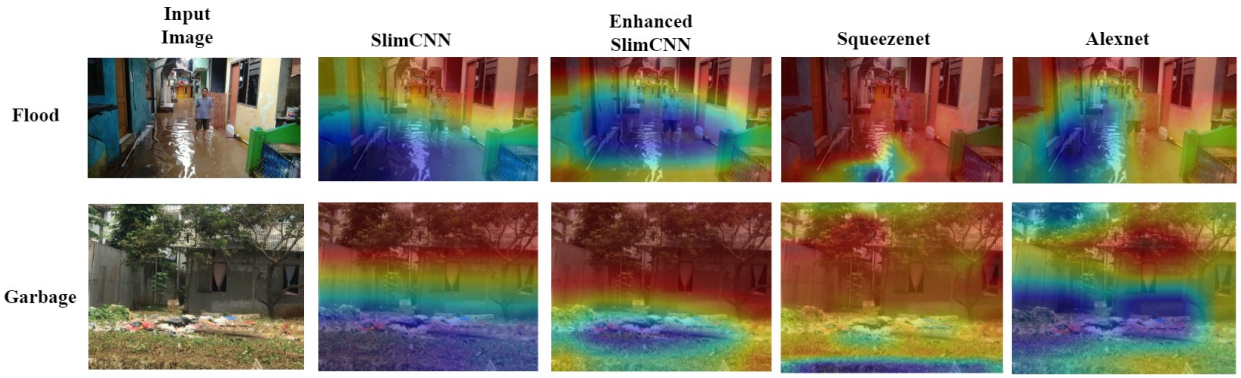

Fig. 5. CAM results for all models. Zoom in for a better resolution.

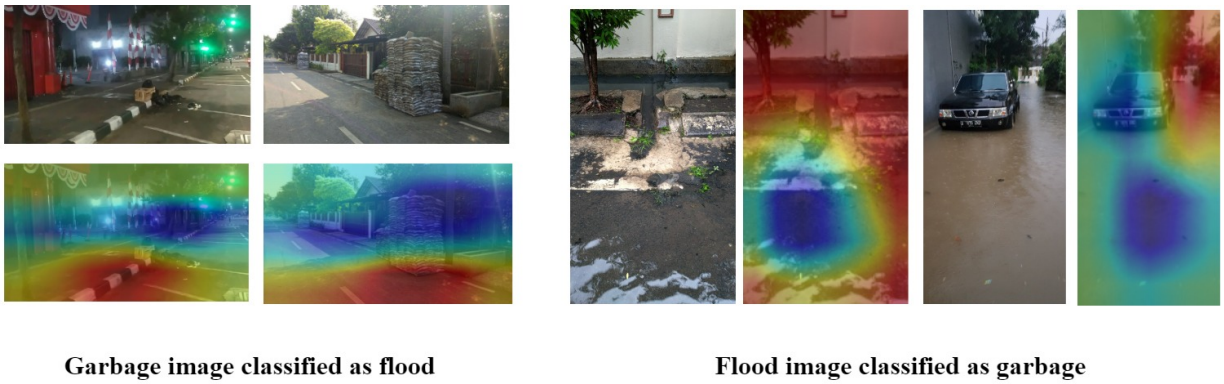

Fig. 6. Incorrect classification on the testing image. Zoom in for a better resolution.

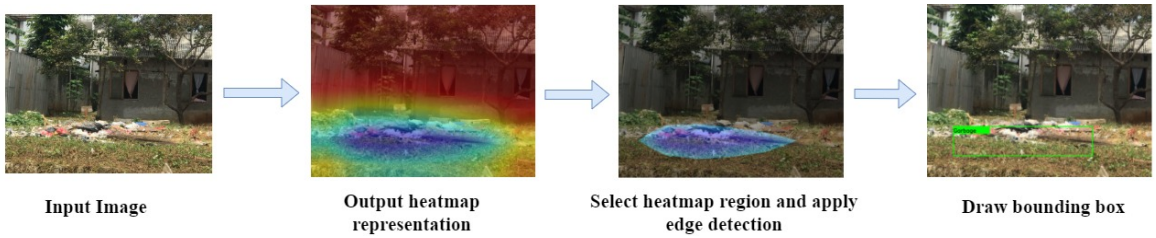

Fig. 7. The procedure of drawing bounding box from heatmap representation. Zoom in for a better resolution.

A. Dataset

The image dataset that we used is obtained from the Jakarta Smart City database (the JAKI database). The dataset consists of two variables. The first variable is the categories of the classes, and the second variable is the report URL on JAKI applications from January 1st, 2020, until December 31st, 2020. In this research, we are only focusing on flood and garbage categories. Table II shows some details of the dataset from the JAKI database.

The dataset fetches from the JAKI database contain some misleading images, such as selfies and blur images. This condition will reduce the accuracy of the model since the contents are outside of the context images. Therefore, we select and clean the dataset manually, containing images of flood and garbage.

We divide them into training, validation, and testing from 2828 clean images of flood and 3796 clean images of garbage. Dividing the data into three subsets is necessary because training data is implemented to fit the model. At the same time, validation data is used to validate the model, and a testing set is used to test the model. This paper divides the dataset using the ratio of 0.64 for training, 0.2 for validation, and 0.16 for testing. Figure 3 shows some examples of the dataset.

B. Exploration on Enhanced SlimCNN

This study uses the Enhanced SlimCNN model to see the effect of adding depth on regular SlimCNN with its accuracy.

Later, we compare the Enhanced SlimCNN with other models, for example, Alexnet and Squeezenet. Table III and Figure 4 shows the parameters and receiver operating characteristic (ROC) curve for all models.

Fig. 8. Bounding box visualization where the green color is the prediction bounding box and the red color is the ground truth. Zoom in for a better resolution.

From Figure 4, we can see that adding some layers to SlimCNN will increase its classification performance because CNN can extract more features. However, increasing the number of features will also increase the storage size and the number of model parameters, as seen in Table III.

Additionally, an interesting result happens between Alexnet and Enhanced SlimCNN, where Alexnet classification performance is lower than SlimCNN. The reason for this is located on the skip connections that Enhanced SlimCNN used. This skip connection prevents the model's accuracy from getting saturated as the network layer goes into more details. Another benefit is that skip connections give an additional path for the gradient in backpropagation; thus, it helps the model converge quickly [18].

To see the localisation of its classes from all models, we use the CAMs method here. The results are represented in the heatmap region, where the blue colour presents the object inside the classes. The red colour depicts the object outside the classes. This result can be seen in Figure 5. From Figure 5, we see that the Enhanced SlimCNN gives a better localisation result compared to regular SlimCNN since the heatmap correctly covers the location of the classes. The effect of adding another layer on SlimCNN allows the model to capture more features. In addition, the CAMs results from Squeezenet and Alexnet seem only to highlight a small region of the object's classes. In contrast, the Enhanced SlimCNN could localise the whole region of the object's classes.

However, some objects are detected as the wrong classes. As an example, in Figure 6, the image of garbage is detected as a flood image, so does the image of the flood, which is detected as a garbage image. Since the amount of our dataset is still limited, the Enhanced SlimCNN model classifies the image improperly.

We could extend our heatmap representation into bounding box visualisation. To convert into bounding box visualisation, we select the most region that the heatmap representation covers. Next, we find the contour of the region by applying edge detection on it. Later, we draw the bounding box over the selected contour. This procedure and the bounding box result can be seen in Figure 7 and Figure 8.

V. Conclusion and Future Work

This paper has analyzed several types of deep learning models, namely Alexnet, Squeezenet, and SlimCNN. We also modify the regular SlimCNN model by adding one convolutional layer and 4 Slim modules.

The experiments indicate that Alexnet, Squeezenet, and SlimCNN do not perform exceptionally well on the flood and garbage images. On the other hand, the enhancements of SlimCNN provide promising results. The results show that increasing the depth of SlimCNN will increase its classification performance. Moreover, using skip connections on SlimCNN also improves the classification accuracy.

In addition, to visualize our results, we use CAMs to help in understanding what CNN sees. In the experiments, the Alexnet, Squeezenet, and SlimCNN fail to localize the presented object in the image. However, adding a convolution layer and slim module as in Enhanced SlimCNN gives better localization results because the enhancements can capture more features due to the additional layers.

We will add more datasets into the training and testing set to reduce its miss classification in future work. We also want to improve our model by implementing visual commonsense reasoning techniques [26]. This technique implements new tasks and large-scale data sets to the visual understanding of the level of cognition. By doing that, it can increase the accuracy and understanding of the model towards the image reports.

We hope that the image processing method can further advance and simplify reporting automation in the JAKI application from our research efforts. Furthermore, the research that we have created and written in this paper is still under development for the JAKI application system owned by the Jakarta provincial government. Therefore, we expect that every report submitted throughout the JAKI application can be processed automatically and recommended based on its categories.

Acknowledgment

The contents are only the authors' views, not the Provincial Government of DKI Jakarta. We declare there are no funds and conflicts of interest. The author would also like to give special thanks to Jakarta Smart City that provides the JAKI image database.

References

[1] James Manyika, Michael Chui, Brad Brown, Jacques Bughin, Richard Dobbs, Charles Roxburgh, and Angela Byers. Big data: The next frontier for innovation, competition, and productivity. 05 2011.

[2] Luís Bettencourt. The uses of big data in cities. *Big Data*, 2:12–22, 03 2014.

[3] Ankit Kumar Sharma and Hassan Foroosh. Slim-cnn: A light-weight CNN for face attribute prediction. *CoRR*, abs/1907.02157, 2019.

[4] B. Zhou, A. Khosla, Lapedriza. A., A. Oliva, and A. Torralba. Learning Deep Features for Discriminative Localization. *CVPR*, 2016.

[5] Farhana Sultana, Abu Sufian, and Paramartha Dutta. Advancements in image classification using convolutional neural network. *CoRR*, abs/1905.03288, 2019.

[6] Loris Nanni, Stefano Ghidoni, and Sheryl Brahnam. Handcrafted vs. non-handcrafted features for computer vision classification. *Pattern Recognition*, 71:158–172, 2017.

[7] T. Ojala, M. Pietikainen, and D. Harwood. Performance evaluation of texture measures with classification based on kullback discrimination of distributions. In *Proceedings of 12th International Conference on Pattern Recognition*, volume 1, pages 582–585 vol.1, 1994.

[8] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Chem. Biol. Drug Des.*, 297:273–297, 01 2009.

[9] Xiaofeng Fu and Wei Wei. Centralized binary patterns embedded with image euclidean distance for facial expression recognition. In *Proceedings of the 2008 Fourth International Conference on Natural Computation - Volume 04*, ICNC '08, page 115–119, USA, 2008. IEEE Computer Society.

[10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, page 1097–1105, Red Hook, NY, USA, 2012. Curran Associates Inc.

[11] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[12] Hieu Pham, Zihang Dai, Qizhe Xie, Minh-Thang Luong, and Quoc V. Le. Meta pseudo labels, 2021.

[13] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[14] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

[15] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *arXiv e-prints*, page arXiv:2010.11929, October 2020.

[16] Neha Sharma, Vibhor Jain, and Anju Mishra. An analysis of convolutional neural networks for image classification. *Procedia Computer Science*, 132:377–384, 01 2018.

[17] Xavier Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. *Journal of Machine Learning Research - Proceedings Track*, 9:249–256, 01 2010.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[19] Forrest N. Iandola, Matthew W. Moskewicz, Khalid Ashraf, Song Han, William J. Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <1mb model size. *CoRR*, abs/1602.07360, 2016.

[20] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015.

[21] Abien Fred Agarap. Deep learning using rectified linear units (relu). *CoRR*, abs/1803.08375, 2018.

[22] François Chollet. Xception: Deep learning with depthwise separable convolutions. *CoRR*, abs/1610.02357, 2016.

[23] Min Lin, Qiang Chen, and Shuicheng Yan. Network In Network. *arXiv e-prints*, page arXiv:1312.4400, December 2013.

[24] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[25] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[26] Rowan Zellers, Yonatan Bisk, Ali Farhadi, and Yejin Choi. From recognition to cognition: Visual commonsense reasoning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.