



ASSESSMENT OF LEARNING: SHUFFLENETV2

A PREPRINT

 **Delvina Wongsono - 2440034050***
Department of Computer Science
Bina Nusantara University
Jakarta Barat, Cengkareng
delvina.wongsono@binus.ac.id

 **Devin Augustin - 2440094352**
Department of Computer Science
Bina Nusantara University
Jakarta Barat, Cengkareng
devin.augustin@binus.ac.id

 **Rico Frenaldi Tokanto - 2440114373**
Department of Computer Science
Bina Nusantara University
Jakarta Barat, Cengkareng
rico.tokanto@binus.ac.id

January 19, 2023

ABSTRACT

Convolutional neural network merupakan jenis neural network yang digunakan untuk komputasi gambar seperti image recognition. Arsitektur neural network saat ini berkembang dengan sangat cepat di mana kecepatan komputasi dan jumlah memori yang dibutuhkan sangat mempengaruhi desain arsitektur. ShuffleNet V2 merupakan sebuah arsitektur baru yang dibangun dengan mempertimbangkan hal-hal di atas. Dengan menggunakan arsitektur ShuffleNet V2, eksperimen akan dilakukan terhadap dataset Flowers Recognition. Hasil yang didapatkan dari segi akurasi tergolong kurang baik namun jika dalam hal komputasi ShuffleNet V2 dapat tergolong cukup cepat.

Keywords ShuffleNetV2 · Convolutional Neural Networks · MobileNet · DenseNet · ICGV2

1 Introduction

Convolution Neural Network (CNN) merupakan salah satu kelas dari deep learning yang mampu melakukan pengenalan gambar serta klasifikasi gambar, CNN pertama kali ada di publik dengan arsitektur LeNet [1] pada tahun 1998, namun CNN tidak terlalu mendapatkan perhatian atau kurang sukses karena keterbatasan yang ada, antara lain dataset uji coba kecil, komputer yang lambat dan salah dalam menerapkan fungsi non linear.

Arsitektur Convolution Neural Network (CNN) itu sendiri telah berevolusi dari tahun ke tahun menjadi lebih akurat dan cepat, sejak karya dari AlexNet [2] berhasil menjadi pemenang ImageNet Competition, yaitu berupa kompetisi untuk klasifikasi dan deteksi citra yang terdiri dari jutaan citra dengan puluhan ribu kelas. Arsitektur AlexNet terdiri dari total 25 layer dengan 9 convolutional layer dan 60 juta parameter. Kesuksesan yang dibawa oleh AlexNet ini diikuti dengan munculnya arsitektur lain dari CNN, seperti VGG [3], GoogleNet [4], ResNet [5], ResNeXt [6], dan pencarian arsitektur netral otomatis [7, 8, 9].

Namun, perlu diketahui bahwa selain akurasi, kompleksitas perhitungan juga merupakan pertimbangan yang penting. Dalam pekerjaan di dunia nyata sering bertujuan untuk mendapatkan akurasi terbaik dibawah budget komputasi yang terbatas, yang diberikan oleh target platform (misalnya perangkat keras) dan aplikasi skenario (misalnya auto driving requires low latency). Ini memotivasi serangkaian pekerjaan untuk desain arsitektur ringan dan pertukaran akurasi kecepatan yang lebih baik, termasuk ShuffleNet V2 [10], MobileNet V2 [11], MobileNet [12], DenseNet [13], IGCv2 [14], and Xception

*Deep Learning: Assesment of Learning, School Of Computer Science, Bina Nusantara University.

Pada paper ini, permasalahan utama yang dihadapi yaitu ingin membentuk sebuah arsitektur yang memiliki tingkat kinerja bagus, memiliki tingkat akurasi yang cukup tinggi, serta memiliki komputasi yang ringan dan cepat untuk menganalisis serta memprediksi dataset berupa Image. Oleh karena itu, untuk mengatasi beberapa permasalahan tadi kami memakai model ShuffleNet V2 yang dikenal sangat efisien pada saat melakukan proses komputasi, hal ini dikarenakan ShuffleNet V2 mampu memuat lebih banyak saluran peta fitur yang dapat membantu dalam memproses lebih banyak informasi yang penting.

2 Previous Research

Judul	Peneliti	Metode	Kesimpulan	Perbedaan
MobileNetV2: Inverted Residuals and Linear Bottlenecks	Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, Liang-Chieh Chen (2019)	Metode Bottleneck depth-separable. Arsitekturnya berisi lapisan Fully Convolutional dengan 32 filter, diikuti dengan 19 lapisan Bottleneck residual.	Metode Linear Bottleneck kurang kuat daripada model dengan non-linear. Hal ini dikarenakan aktivasi selalu dapat beroperasi dalam rezim linier dengan perubahan bias dan penskalaan yang sesuai.	Dengan menggunakan COCO dataset, arsitektur yang digabungkan dengan model deteksi SSDLite menghasilkan komputasi 20x lebih sedikit dan parameter 10x lebih sedikit daripada YOLOv2.
MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications	Howard Menglong, Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, Hartwig Adam (2017)	Menggunakan metode Depthwise Separable Convolutions yang dipisahkan dengan dua hyperparameter global sederhana dan secara efisien menukar antara latensi dan akurasi.	Hasilnya jika dibandingkan dengan GoogleNet dan VGG16 dalam hal akurasi MobileNet lebih unggul daripada GoogleNet namun masih kalah dengan VGG16, jika dibandingkan dengan SqueezeNet dan AlexNet dalam hal akurasi, MobileNet lebih unggul dari kedua model tersebut.	Menggunakan dataset COCO dan perbedaannya terletak pada saat merancang arsitektur untuk kinerja yang lebih baik dalam penggunaan seluler serta akurasi dan kecepatan Using COCO.
Densely Connected Convolutional Networks	Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger (2018)	Menggunakan metode ResNet dan metode DenseNet.	DenseNet meningkatkan akurasi secara konsisten dengan memakai banyak parameter, tanpa adanya overfitting. Cocok digunakan untuk tugas visi komputer yang dibangun di atas fitur konvolusi.	Menggunakan dataset CIFAR10 dan dataset SVHN, serta meningkatkan arsitektur dengan menggunakan model kompak dengan menghubungkan setiap lapisan.
IGCV2: Interleaved Structured Sparse Convolutional Neural Networks	Guotian Xie, Jingdong Wang, Ting Zhang, Jianhuang Lai, Richang Hong, Guo-Jun Qi (2018)	Membandingkan model IGCV2 dengan model - model lainnya.	Hasil empiris menunjukkan keunggulan dibandingkan model MobileNetV1 dan MobileNetV2 serta menunjukkan bahwa jaringan dengan ukuran model yang lebih kecil mencapai kinerja yang serupa jika dibandingkan dengan struktur jaringan lainnya.	Menggunakan dataset CIFAR10 dan Tiny ImageNet, serta merancang arsitektur CNN yang efisien dengan menghilangkan redundansi pada kernel convolution.

Xception: Deep Learning with Depthwise Separable Convolutions	Francois Chollet (2017)	Menggunakan metode Depthwise Convolution dan diikuti oleh Pointwise Convolution	Pengamatan ini mengusulkan penggantian modul Inception dengan konvolusi yang dapat dipisahkan dalam arsitektur visi komputer saraf. Kinerja pada dataset ImageNet dan JFT juga meningkat dibandingkan dengan Inception V3.	Menggunakan dataset ImageNet dan metode Depthwise Separable convolution
---	-------------------------	---	--	---

Table 1: Previous Research

Penelitian dilakukan oleh Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, Liang-Chieh Chen, menggunakan dataset COCO sebagai eksperimennya. Dalam makalah ini mereka menggambarkan arsitektur seluler baru, MobileNetV2 menggunakan Bottleneck depth-separable convolution dengan metode residual [11]. Arsitekturnya berisi lapisan fully convolutional dengan 32 filter dan diikuti oleh 19 lapisan residual bottleneck, juga arsitektur yang digabungkan dengan model deteksi SSDLite yang komputasinya 20x lebih sedikit dan parameter 10x lebih sedikit daripada YOLOv2. Hasilnya yaitu model Linear Bottleneck kurang kuat jika dibandingkan dengan model non-linearitas, karena aktivasi selalu dapat beroperasi dalam rezim linier dengan perubahan bias dan penskalaan yang sesuai. Bagaimanapun Linear Bottleneck kemacetan Linier meningkatkan kinerja dan memberikan dukungan bahwa nonlinier menghancurkan informasi dalam ruang berdimensi rendah.

Penelitian dilakukan oleh Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, Hartwig Adam, mereka menggunakan dataset COCO sebagai eksperimennya. Mereka menghadirkan kelas model efisien yang disebut sebagai MobileNets dan menggunakan convolution yang dapat dipisahkan secara mendalam dengan dua hyperparameter global sederhana yang secara efisien menukar antara latensi dan akurasi [12]. Arsitekturnya dirancang untuk kinerja yang lebih baik dalam penggunaan seluler, akurasi dan kecepatan. Hasilnya jika dibandingkan dengan GoogleNet dan VGG16 dalam hal akurasi MobileNet lebih unggul dari GoogleNet namun masih kalah dengan VGG16, jika dibandingkan dengan SqueezeNet [15] dan AlexNet dalam hal akurasi, MobileNet lebih unggul dari kedua model tersebut.

Penelitian dilakukan oleh Gao Huang, Zhuang Liu, Laurens van der Maaten, mereka menggunakan dataset CIFAR10[16], dataset SVHN serta dataset ImageNet. Dalam makalah ini mereka menggunakan Dense Convolutional Network (DenseNet) yang menghubungkan setiap lapisan ke setiap lapisan lainnya secara feed-forward, kemudian dibandingkan dengan ResNet [13]. Hasilnya antara DenseNet dan ResNet jika menggunakan dataset ImageNet maka uji kesalahan ResNet sedikit lebih baik daripada DenseNet.

Penelitian dilakukan oleh Guotian Xie dan yang lainnya, mereka menggunakan dataset CIFAR10 dan Dataset Tiny ImageNet. Mereka merancang arsitektur CNN yang efisien dengan menghilangkan redundansi pada kernel convolution dan menghadirkan blok Interleaved Structured Sparse Convolution (IGCV2) guna menyusun kernel yang padat [14]. Hasilnya menunjukkan keunggulan dibandingkan MobileNetV1 dan MobileNetV2 dan menunjukkan bahwa jaringan dengan ukuran model yang lebih kecil mencapai kinerja yang sama jika dibandingkan dengan struktur jaringan lainnya.

Penelitian dilakukan oleh Francois Chollet dengan menggunakan ImageNet dataset dan metode Depthwise Separable Convolution [17]. Penelitian ini menunjukkan bagaimana convolution dan depthwise separable convolutions terletak pada kedua ekstrem spektrum diskrit, dengan modul Inception [18] menjadi titik perantara di antaranya. Pengamatan ini mengusulkan penggantian modul Inception dengan convolution yang dapat dipisahkan secara mendalam dalam arsitektur visi komputer saraf.

3 Methodology or Architecture Deep Learning

3.1 ShuffleNetV2 Architecture

Tabel di bawah menunjukkan arsitektur keseluruhan dari ShufflenetV2.

Layer	Output size	KSize	Stride	Repeat	Ouput Channels			
					0.5X	1X	1.5	2X
Image	224x224				3	3	3	3
Conv1	112x112	3x3	2	1	24	24	24	24
MaxPool	56x56	3x3	2	1	24	24	24	24
Stage2	28x28		2	1	48	116	176	244
Stage2	28x28		1	3	48	116	176	244
Stage3	14x14		2	1	96	232	352	488
Stage3	14x14		1	7	96	232	352	488
Stage4	7x7		2	1	192	464	702	976
Stage4	7x7		1	3	192	464	702	976
Conv5	7x7	1x1	1	1	1024	1024	1024	1024
GlobalPool	1x1	7x7						
FC					1000	1000	1000	1000
FLOPs					41M	146M	299M	591M
NumberD of Weights					1.4M	2.3M	3.5M	7.4M

Table 2: ShuffleNetV2 Architecture

3.2 Architecture Explanation

3.2.1 ShuffleNet V2

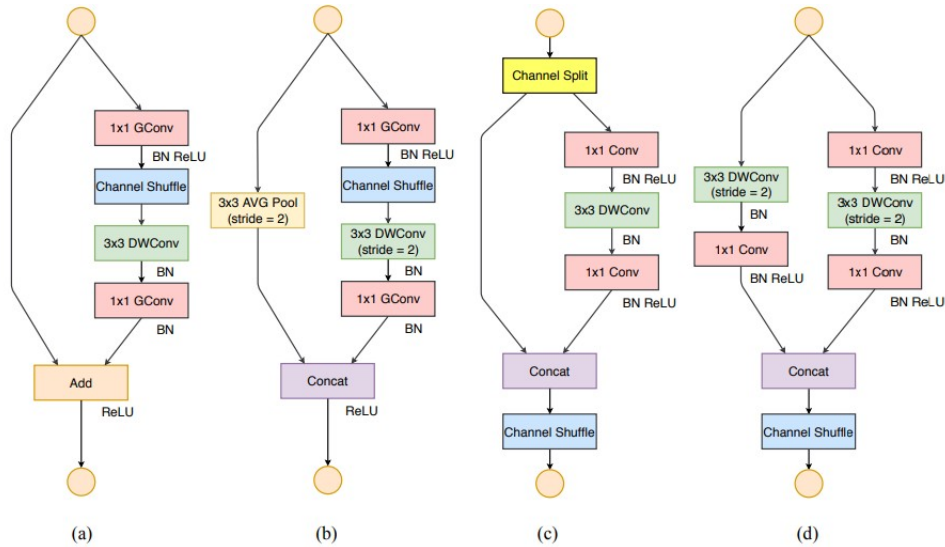


Figure 1: ShuffleNetV2 Architecture

Terdapat Image berukuran 224 x 224 yang terdiri atas 3 filter channel dengan masing - masing Output channel yang sudah tertera di tabel.

Pertama, masuk ke Lapisan Convolution awal (Conv1) [19] dan MaxPool [20], ini merupakan jenis lapisan CNN yang terdiri atas 24 filter channel yang berukuran 3 x 3 dengan nilai langkah (Stride) 2 serta terdapat satu stage shuffle, Output size untuk Conv1 sebesar 112 x 112 sementara untuk MaxPool memiliki Output size sebesar 56 x 56.

Kedua, masuk ke lapisan Stage2 dimana terdiri atas dua stage shuffle (1 dan 3) dengan nilai stride (2 dan 1). Stage2 memiliki Output size sebesar 28 x 28 dan untuk Output channels dapat dilihat pada tabel di atas.

Ketiga, masuk ke lapisan Stage3 dimana terdiri atas dua stage shuffle (1 dan 7) dengan nilai stride (2 dan 1). Stage3 memiliki Output size sebesar 14 x 14 dan untuk Output channels dapat dilihat pada tabel di atas.

Keempat, masuk ke lapisan Stage4 dimana terdiri atas dua stage shuffle (1 dan 3) dengan nilai stride (2 dan 1). Stage4 memiliki Output size sebesar 7 x 7 dan untuk Output channels dapat dilihat pada tabel di atas.

Kelima, yaitu Lapisan Convolution akhir (Conv5) yang berukuran 1 x 1 dengan nilai Stride 1 dan terdiri atas 1 stage shuffle. Lapisan ini memiliki Output size 7 x 7 serta Output channel dapat dilihat pada tabel di atas.

Keenam, dilakukan GlobalPool yang berukuran 7 x 7, sehingga menghasilkan Output size 1 x 1.

Ketujuh, masuk ke Fully Connected [21] (FC) yang menghasilkan nilai Output channel sebesar 1000, selanjutnya dilakukan Floating Point Operations Per seconds (FLOPs), hal ini berfungsi untuk mengukur/menggambarkan berapa banyak operasi yang dibutuhkan untuk menjalankan sebuah model jaringan.

3.2.2 MobileNet V1

Tabel di bawah merupakan arsitektur keseluruhan dari MobileNet.

Type/Stride	Filter Shape	Input Size
Conv/s2	3x3x3x32	224x224x3
Conv dw/s1	3x3x32 dw	112x112x32
Conv/s1	1x1x32x64	112x112x32
Conv dw/s2	3x3x64 dw	112x112x64
Conv/s1	1x1x64x128	56x56x64
Conv dw/s1	3x3x128 dw	56x56x128
Conv/s1	1x1x128x128	56x56x128
Conv dw/s2	3x3x128 dw	56x56x128
Conv/s1	1x1x128x256	28x28x128
Conv dw/s1	3x3x256 dw	28x28x256
Conv/s1	1x1x256x256	28x28x256
Conv dw/s2	3x3x256 dw	28x28x256
Conv/s1	1x1x256x512	14x14x256
5x Conv dw/s1	3x3x512 dw	14x14x512
5x Conv/s1	1x1x512x512	14x14x512
Conv dw/s2	3x3x512 dw	14x14x512
Conv/s1	1x1x512x1024	7x7x512
Conv dw/s2	3x3x1024 dw	7x7x1024
Conv/s1	1x1x1024x1024	7x7x1024
Avg Poll/s1	Pool 7x7	7x7x1024
FC/s1	1024x1000	1x1x1024
Softmax/s1	Classifier	1x1x1000

Table 3: MobileNet V1 Architecture

Pada arsitektur MobileNet terdapat Input size (224x224x3) dimana dilakukan Convolution dengan Stride 2 dan Filter Shape sebesar (3x3x3x32), setelah itu dilakukan Depthwise Convolution dengan stride 1 yang memperkecil Input size menjadi (112x112x32) dengan Filter Shape (3x3x32). Proses ini dilakukan berulang kali guna memperkecil dimensi gambar, setelah Input size menjadi sebesar (7x7x1024) maka dilakukan Average Pooling yang berukuran 7x7 dengan Stride 1. Selanjutnya, masuk ke lapisan Fully Connected Layer dengan Stride 1, filter shape menjadi 1024x1000 dan Input size menjadi 1x1x1024. Setelah itu, masuk ke lapisan Softmax yang berguna untuk menghitung probabilitas pada hasil output, yang terjadi di output layer dimana akan diambil nilai probabilitas yang paling besar sebagai hasil prediksi.

4 Result and Analysis

Pada eksperimen kami, kita menggunakan dataset “Flowers Recognition”[22] yang terdiri dari 4242 gambar bunga dan terdapat 5 jenis, yaitu chamomile, tulip, rose, sunflower, dan dandelion. Setiap jenis bunga memiliki sekitar 800 foto dan memiliki ukuran sebesar 320 x 240 piksel. Sebelum melakukan training terhadap dataset, data akan dibagi dua menjadi train dan validation dengan rasio delapan banding dua [23].

Learning rate yang digunakan adalah 0.1 dengan parameter epoch 30 dan batch size 16. Di sini kami memilih learning rate 0.1 karena jika learning rate terlalu besar, maka akurasi yang dihasilkan akan menjadi buruk dan waktu yang dibutuhkan training akan semakin besar [24]. Berikut adalah hasilnya:

4.1 Training and Validation Loss

Jika dilihat pada grafik di di bawah, dikatakan bahwa hasil akurasi akan bagus apabila mendekati angka 1 dengan kata lain semakin tinggi angkanya, maka hasil semakin baik. Sebaliknya untuk loss, semakin kecil angkanya (mendekati angka 0) maka hasil semakin bagus [25]. Berdasarkan grafik di atas, dapat disimpulkan bahwa tidak terjadinya overfitting maupun underfitting terhadap dataset “Flower Recognition” [26].

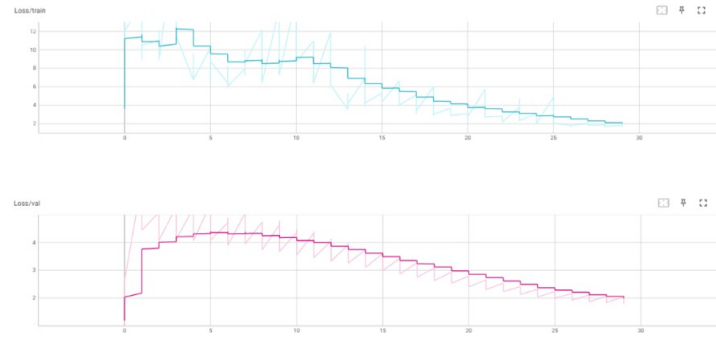


Figure 2: Train and Validation Loss

4.2 Training and Validation Accuracy

Dan juga dapat dilihat bahwa Learning rate yang dihasilkan sebesar 0.1, sementara itu angka kurva pada akurasi hanya sampai di 0.38 dan 0.35.

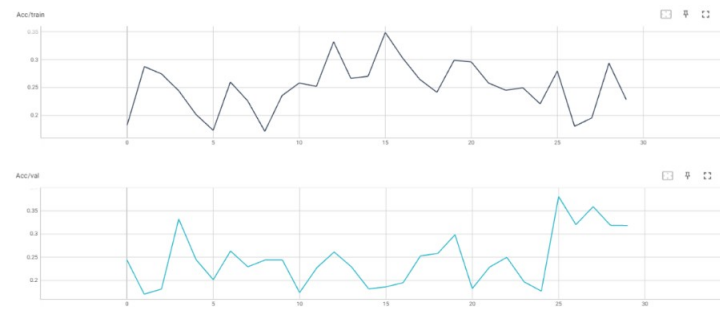


Figure 3: Train and Validation Accuracy

Setelah itu, dilakukan test terhadap model yang telah kami jalankan. Setelah memasukkan beberapa gambar bunga, hasil prediksi yang kami peroleh bisa dikatakan cukup akurat di mana saat gambar tulip dimasukkan, maka model berhasil memprediksi dengan benar dan menghasilkan probabilitas sebesar 0.337 dan untuk gambar daisy dihasilkan probability sebesar 0.995. Hasil juga dapat dilihat pada gambar di bawah ini.

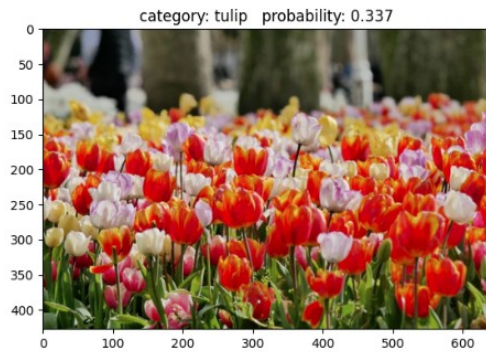


Figure 4: Test One.

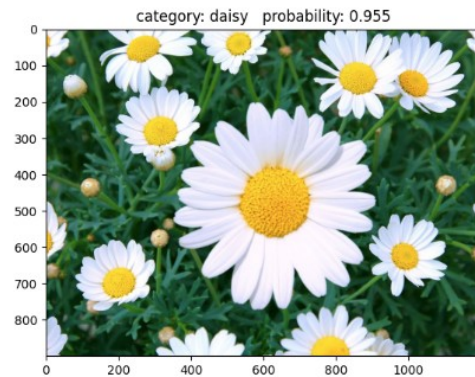


Figure 5: Test Two.

5 Conclusion

Dari hasil analisa yang telah dilakukan terhadap dataset “Flower Recognition” menggunakan code ShuffleNetV2 oleh Lin Feng Lee, hasil yang diperoleh hanya sekitar 0.3 hingga 0.4, hal ini dinilai kurang memuaskan jika dinilai dari segi akurasi. Tetapi, jika dilihat dari nilai loss, hasil yang diperoleh cukup baik hal ini dikarenakan nilainya yang semakin rendah dan mendekati angka 0. Selain itu, waktu komputasi yang dihasilkan juga tidak terlalu lama. Namun, secara keseluruhan dan adanya sedikit peningkatan terhadap kode, arsitektur ini merupakan arsitektur yang bagus untuk diterapkan untuk menganalisis dataset berupa Image.

References

- [1] Ben Ding. Lenet: Lightweight and efficient lidar semantic segmentation using multi-scale convolution attention. 01 2023.
- [2] Sutskever I. Hinton G.E. Krizhevsky, A. Imagenet classification with deep convolutional neural networks. in: Advances in neural information processing systems.
- [3] Zisserman A. Simonyan, K. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- [4] Liu W. Jia Y. Sermanet P. Reed S. Anguelov-D. Erhan D. Vanhoucke V. Rabinovich A. et al Szegedy, C. Going deeper with convolutions, *cvpr*. 2015.
- [5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi:10.1109/CVPR.2016.90.
- [6] Girshick R. Doll’ar P. Tu Z. He K. Xie, S. Aggregated residual transformations for deep neural networks. in: Computer vision and pattern recognition (cvpr). *2017 IEEE Conference on, IEEE (2017)* 5987–5995, 2017.
- [7] Vasudevan V. Shlens J. Le Q.V. Zoph, B. Learning transferable architectures for scalable image recognition. *arXiv preprint arXiv:1707.07012*, 2017.
- [8] Zoph B. Shlens J. Hua W. Li L.J. Fei-Fei L. Yuille A. Huang J. Murphy K. Liu, C. Progressive neural architecture search. *arXiv preprint arXiv:1712.00559*, 2017.
- [9] Aggarwal A. Huang Y. Le Q.V. Real, E. Regularized evolution for image classifier architecture search. *arXiv preprint arXiv:1802.01548*, 2018.
- [10] Zhang X. Zheng H. Sun J. Ma, N. Shufflenet v2: Practical guidelines for efficient cnn architecture design. 2018.
- [11] Howard A. Zhu M. Zhmoginov A. Chen L.C. Sandler, M. Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation. *arXiv preprint arXiv:1801.04381*, 2018.
- [12] Zhu M. Chen B. Kalenichenko D. Wang W. Weyand-T. Andreetto M. Adam H. Howard, A.G. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

- [13] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, 2017. doi:10.1109/CVPR.2017.243.
- [14] Guotian Xie, Jingdong Wang, Ting Zhang, Jianhuang Lai, Richang Hong, and Guo-Jun Qi. IGCv2: interleaved structured sparse convolutional neural networks. *CoRR*, abs/1804.06202, 2018. URL <http://arxiv.org/abs/1804.06202>.
- [15] Forrest Iandola, Matthew Moskewicz, Khalid Ashraf, Song Han, William Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and textless1mb model size. 02 2016.
- [16] Akwasi Akwaboah. Convolutional neural network for cifar-10 dataset image classification. 11 2019.
- [17] François Chollet. Xception: Deep learning with depthwise separable convolutions, 2016.
- [18] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014. URL <http://arxiv.org/abs/1409.4842>.
- [19] Phil Birch, Navid Rahimi, Peter Overbury, Rupert Young, and Chris Chatwin. Implementations and optimisations of optical conv2d networks designs. 03 2020.
- [20] Somenath Bera and Vimal Shrivastava. Effect of pooling strategy on convolutional neural network for classification of hyperspectral remote sensing images. *IET Image Processing*, 14, 02 2020. doi:10.1049/iet-ipr.2019.0561.
- [21] Sh Shabbeer Basha, Shiv Ram Dubey, Viswanath Pulabaigari, and Snehasis Mukherjee. Impact of fully connected layers on performance of convolutional neural networks for image classification. *Neurocomputing*, 378, 10 2019. doi:10.1016/j.neucom.2019.10.008.
- [22] Alexander Mamaev. Flowers recognition dataset. URL <https://www.kaggle.com/datasets/alxmamaev/flowers-recognition>.
- [23] Adam; Engels Daniel W.; Barry-Straume, Jostein; Tschannen and Edward Fine. An evaluation of training size impact on validation accuracy for optimized convolutional neural networks. *SMU Data Science Review*, 1, 2018.
- [24] D. Wilson and Tony Martinez. The need for small learning rates on large problems. volume 1, pages 115 – 119 vol.1, 02 2001. ISBN 0-7803-7044-9. doi:10.1109/IJCNN.2001.939002.
- [25] Simon Dräger and Jannik Dunkelau. Evaluating the impact of loss function variation in deep learning for classification. 10 2022.
- [26] Jeff Bilmes. Underfitting and overfitting in machine learning. 2020.