Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

# Optimal Training Configurations Of Deep Learning Model Overcome Architectural Barrier

**Shu-Hao Liu**

shliu0607@gmail.com

John Burroughs School https://orcid.org/0009-0007-7655-587X

**Research Article**

**Keywords:** Machine Learning, Biology, Plant Science.

**Posted Date:** November 26th, 2026

**DOI:** https://doi.org/10.21203/rs.3.rs-11847632/v1

**License:** This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

**Additional Declarations:** The authors declare no competing interests.

# Abstract

Comparisons of convolutional neural network architectures for plant disease classification often attribute performance differences to architectural depth alone, ignoring the role of training hyperparameters. Two models have been commonly used for developing deep learning algorithms: VGG16 and GoogLeNet, with GoogLeNet widely considered superior to VGG16. We also consider the fact that they were optimized with different hyperparameters, making them fundamentally different in how they learn and generalize. This study tested the hypothesis that plant diseases exhibit visual features that can be consistently identified by convolutional neural networks, and variations in training hyperparameters significantly influence the reliability of automated disease classification across model architectures. We used a published plant disease dataset composed of 54,303 images in 38 classes which was used to develop a benchmark GoogLeNet model previously. After creating a directly testable GoogLeNet model with consistent hyperparameters as the benchmark and comparing it with our VGG16 model, we found that the accuracy of the two models only differed by < 0.5%. Overall, our study demonstrated the critical role of consistent training configurations in developing deep learning algorithms and suggests that network depth is not the sole factor behind a model's performance.


# Introduction

Deep learning,[1] which involves training models with huge amounts of data using neural networks, has shown immense potential in image classification.[2] Such a capability can be utilized for many fields, including agriculture, where the rapid identification of plant diseases is vital,[3] as some diseases can spread as fast as 9.8 minutes.[4] As a result, around 40% of crops are lost to diseases each year, according to the Food and Agriculture Organization of the United Nations.[5]Indeed, the World Health Organization predicts around 583 million people will be undernourished in 2030.[6] Therefore, there exists a great need to develop methods to identify plant disease efficiently, and deep learning is one promising approach. While there are many deep learning models available for this task,[7]two models were commonly used: VGG16[8] and GoogLeNet.[9]

Released in 2014, VGG16 is a robust convolutional neural network (CNN) consisting of 16 layers, including 13 convolutional layers. Its simple architecture makes it much easier to grasp than GoogLeNet while delivering strong performance.

Published in the same year as VGG16, GoogLeNet is another strong CNN. The two models have fundamental differences. Instead of simply stacking convolution layers, GoogLeNet introduces a unique feature: the Inception module (GoogLeNet contains 9 of them), which allows for the simultaneous extraction of information. Through the innovation of Inception modules, GoogLeNet was able to capture many kinds of patterns in images that would usually be overlooked with a single-sized filter, as in the case of VGG16. Moreover, GoogLeNet's design addressed problems such as computational costs and choosing the best filter size with its fewer parameters and use of different filter sizes in parallel.

The GoogLeNet model is often reported to deliver better results than VGG16. In the 2014 ImageNet Large Scale Visual Recognition Challenge,[10] designed to benchmark and compare top computer vision models, GoogLeNet scored a top-5 score (defined as the percentage of times the correct label for an image wasn't among the model's top five predicted categories) of 6.7%, while VGG16 scored 7.3%, demonstrating that GoogLeNet surpassed VGG16 in classification accuracy. Furthermore, a study published in the Indian Journal of Science and Technology showed that GoogLeNet outperformed VGG16 by a substantial 12.1% in F1 score (an accuracy metric designed to provide a balanced measure of a model's performance) on lung cancer classification[11] and another article in the Journal of Physics: Conference Series demonstrated that VGG16 performed worse than GoogLeNet on the popular CIFAR100 image dataset.[12] It is generally believed that GoogLeNet's deeper architecture allows for better learning, as it can extract more complex and hierarchical patterns than VGG16 due to its unique design.[9] However, many prior comparisons optimize each architecture independently, making it difficult to determine whether performance differences arise from architectural design or from training configuration. As a result, it is unclear whether GoogLeNet's better performance is entirely due to its architecture.

When training a deep learning model, a crucial step is to tune the model's hyperparameters, which are the external settings that determine how a model will run (the batch size, epoch count, optimizer, etc.). With optimal hyperparameters, a model can perform well, while suboptimal configurations may prevent it from reaching its full potential.[13] Furthermore, when evaluating multiple models on performance, these models were often optimized separately.[14] This leads to the key question: whether optimized hyperparameters could modulate the accuracy of the model. Despite VGG16 usually being outperformed by GoogLeNet, we believe that its straightforward architecture and ability to introduce greater non-linearity may allow its capabilities to be fully realized. We therefore hypothesize that plant diseases exhibit visual features that can be consistently identified by convolutional neural networks, and variations in training hyperparameters significantly influence the reliability of automated disease classification across model architectures.

# Methods

### *Dataset*

The PlantVillage dataset was used for the training and validation of the algorithm. The dataset included pre-defined train-test splits. The dataset contained 54,303 images across 38 classes.

### *Benchmark model*

The benchmark model,[20] as developed by Penn State University (PSU), utilized an 80-20 train-test split and a GoogLeNet base model, with the entire base model unfrozen and fine-tuned on the PlantVillage dataset.

### *Model development and implementation*

The images in the dataset were first rescaled to dimensions of (224,224,3). A VGG16 model with weights from ImageNet,[16] without its original classification layers, was then used as the base model. All layers in the VGG16 model were unfrozen and fine-tuned to adapt to the dataset. An average pooling layer was then used to reduce spatial dimensions (Figure 1). Training was done on an NVIDIA P100 GPU.

**Figure 1.** The architecture for transfer learning with VGG16 for plant disease classification. A sequential diagram of the model's architecture, which includes a rescaling layer, a VGG16 base model (unfrozen and fine-tuned), an average pooling layer to increase efficiency, and an output layer with Softmax activation that makes the model predictions. The architecture diagram was generated using Netron[24] from a .keras file.

To ensure a fair and consistent comparison with the benchmark model, the following hyperparameters were used:

Optimizer: Stochastic Gradient Descent; Base learning rate: 0.005; Learning rate decay: by a factor of 10 every 10 epochs; Momentum: 0.9; Weight decay: 0.0005; Gamma: 0.1; Epochs: 30; Batch size: 64.

### *Statistical analysis*

The accuracy of the model was evaluated using an F1 score, computed through TensorFlow((21))'s built in metric, and rounded to four decimal places. The F1 score was chosen to evaluate model performance as it measures not only basic accuracy, but the model's ability to balance precision and recall (two metrics for evaluating model performance), which provides a more accurate metric for imbalanced datasets. A 95% confidence interval was computed using bootstrap resampling to determine the range in which the true value of the F1 score lies. Bootstrap resampling involves feeding various samples from the validation set into a model (with replacement) and then calculating the model's F1 score. After feeding 1000 samples into the model, the mean F1 score and the lower and upper bounds of the confidence interval were calculated. Additionally, as the benchmark model was not directly testable, we applied the same bootstrap resampling method to the replicate model to calculate a confidence interval to ensure a better comparison with our model. Bootstrap resampling was done using NumPy[22] and Scikit-learn.[23]

## Results and Discussion

### *VGG16 model setup*

We designed and evaluated a VGG16 model with the same training settings against a benchmark GoogLeNet model trained on the identical PlantVillage dataset[15] (Figure 1). In our model, each image was first rescaled to a size of (224,224,3) and passed through an unfrozen VGG16 base model with weights from ImageNet.[16] An average pooling layer was also used to reduce spatial dimensions. We

used the same optimizer, base learning rate, learning rate decay, momentum, weight decay, epoch count, and batch size as the benchmark model. Training was done on an NVIDIA P100 GPU.

***Performance of the VGG16 model against the benchmark model***

Matplotlib[17] was used to compute a high-level visualization of the VGG16 model's accuracy breakdown by class (Figure 2). For the validation set, the accuracy (F1 score) for the VGG16 model was 99.19% (98.7% - 99.24%, 95% confidence interval). The model's F1 score progression on the validation set during training is shown in Figure 3. However, as the original benchmark model was not open-sourced in a way that would allow us to test it, we replicated the model (training progression is shown in Figure 4) and obtained an F1 score of 99.54% on the validation set, (99.37% - 99.71%, 95% confidence interval), which was higher than the original F1 score of the benchmark model (99.34%) (Table 1). Despite this, the VGG16 model's F1 score was 0.15% from the benchmark model's reported F1 score and 0.35% from the replica's F1 score. Considering the upper bound of the replica's confidence interval (99.71%), the difference would be 0.52%.

**Figure 2.** VGG16 model accuracy breakdown by class. The blue bars represent total samples in any given specific class of plant disease, while the red bars represent the total samples in that class that the model predicted correctly. Computed with Matplotlib, a Python library for graph drawing.

**Figure 3.** F1 score graph of the VGG16 model on the validation set during training. On the x-axis is the number of training epochs, while on the y-axis the corresponding F1 score is shown. Computed with Matplotlib.

**Figure 4.** F1 score graph of the GoogLeNet model on the validation set during training. On the x-axis is the number of training epochs, while on the y-axis the corresponding F1 score is shown. Computed with Matplotlib.

**Table 1.** F1 scores and confidence intervals between the three models.

| Model | F1 score (%) | 95% confidence interval (%) |
|---|---|---|
| VGG16 | 99.19 | 98.7 - 99.24 |
| GoogLeNet (replica) | 99.54 | 99.37-99.71 |
| GoogLeNet (benchmark) | 99.34 | N/A |

By training VGG16 with the same hyperparameters as a benchmark GoogLeNet model, we obtained findings that supported our hypothesis that plant diseases exhibit visual features that can be consistently identified by convolutional neural networks, and variations in training hyperparameters significantly influence the reliability of automated disease classification across model architectures. We also made a replica of the benchmark model, as the model itself was not available, and computed its confidence interval along with that of our model, which showed that the confidence intervals were in

proximity (differing by only 0.13%). This provided an alternative perspective into the prevalent consensus that GoogLeNet performs better than VGG16.[18] Our findings suggest that under identical hyperparameters, VGG16 can achieve comparable performance compared to GoogLeNet, despite its simpler architecture. One limitation of our study is that we only trained our model on preprocessed leaf images, that is, images of plant leaves taken in consistent lighting and background conditions, instead of leaves “out in the wild”, taken in different lighting conditions and backgrounds. This was done in order to ensure a consistent comparison with the benchmark, which used the same set of images. Future work could involve training the models with a real-life dataset of plant images, which may potentially yield different results, as the architectures of VGG16 and GoogLeNet were designed to capture different types of image features, with VGG16 having a

straightforward architecture consisting of sequential convolution layers, and GoogLeNet utilizing parallel convolutions to capture different types of patterns among images. Such a dataset would potentially reveal overlooked strengths and weaknesses of both models, as most image datasets try to provide images taken in controlled conditions as part of a broader trend by the machine learning community to ensure high-quality datasets[19], and real-life images may present different challenges than lab-processed ones, such as the challenge of inconsistent backgrounds and lighting. Another limitation of this experiment is the fact that we were unable to test the benchmark model directly as the code was not publicly accessible. Therefore, these conclusions apply specifically to high-quality, controlled image datasets and may not directly generalize to agricultural environments without further validation.

# Conclusion

Overall, a VGG16 model trained using the same configurations as a GoogLeNet benchmark model achieved a comparable F1 score. This challenges the assumption that the depth of networks alone directly correlates to better performance and suggests that a more holistic approach that accounts for training configurations as well as architecture is needed to accurately and fairly evaluate the performance of models.

# Declarations

### AI Acknowledgement

The author(s) declare(s) that no artificial intelligence (AI) tools were used in the preparation of this work.

# References

1. LeCun Y, Bengio Y, Hinton G, Deep (2015) learning. Nature 521, 436–444. https://doi.org/10.1038/nature14539

2. Krizhevsky A, Sutskever I, Hinton GE (2017) ImageNet classification with deep convolutional neural networks. Commun ACM 60:84–90. https://doi.org/10.1145/3065386
3. Buja I, Sabella E, Monteduro AG, Chiriacò MS, De Bellis L, Luvisi A, Maruccio G (2021) Advances in plant disease detection and monitoring: from traditional assays to in-field diagnostics. Sensors 21:2129. https://doi.org/10.3390/s21062129
4. DuPont ST, Gugino B, Swackhamer E (2025) Plant disease basics and diagnosis. Penn State Ext, https://extension.psu.edu/plant-disease-basics-and-diagnosis
5. Food and Agriculture Organization of the United Nations (FAO). About plant production and protection. Food and Agriculture Organization of the United Nations (2025) https://www.fao.org/plant-production protection/about/en
6. World Health Organization (WHO). Hunger numbers stubbornly high for three consecutive years as global crises deepen: UN report. World Health Organization News Release (2024) https://www.who.int/news/item/24-07-2024-hunger-numbers-stubbornly-high-for-three-consecutive-years as-global-crises-deepen-un-report
7. Dong X, Wang Q, Huang Q, Ge Q, Zhao K, Wu X, Wu X, Lei L, Hao G (2023) PDDD-PreTrain: a series of commonly used pre-trained models support image-based plant disease diagnosis. Plant Phenomics 5:0054. https://doi.org/10.34133/plantphenomics.0054
8. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409 1556. https://doi.org/10.48550/arXiv.1409.1556
9. Szegedy C, Liu W, Jia Y, Sermanet P, Reed SE, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A (2014) Going deeper with convolutions. arXiv preprint arXiv:1409.4842. https://doi.org/10.48550/arXiv.1409.4842
10. Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, Berg AC (2015) Fei-Fei, L. ImageNet large scale visual recognition challenge. Int J Comput Vis 115:211–252. https://doi.org/10.1007/s11263-015-0816-y
11. Sridevi CS, Kannan M (2025) Convolutional neural network architecture-Inception (GoogLeNet) for deep architected learning-assisted lung cancer classification in computed tomography images. Indian J Sci Technol 18:504–516. https://doi.org/10.17485/IJST/v18i7.3086
12. Zheng YY, Huang HX, Chen JM (2024) Comparative analysis of various models for image classification on CIFAR–100 dataset. J Phys Conf Ser 2711:012015. https://doi.org/10.1088/1742-6596/2711/1/012015
13. Arnold L, Biedebach A, Küpfer A, Neunhoeffer M (2024) The role of hyperparameters in machine learning models and how to tune them. Polit Sci Res Methods 12:841–848. https://doi.org/10.1017/psrm.2023.61
14. Grillo B, Kontorinaki M, Sammut F (2025) A comparative analysis of hyperparameter effects on CNN architectures for facial emotion recognition. In Proceedings of the 14th International Conference on Pattern Recognition Applications and Methods; ; Vol. 1, pp 587–596. https://doi.org/10.5220/0013146900003905

15. Hughes DP, Salathé M An open access repository of images on plant health to enable the development of mobile disease diagnostics. arXiv preprint arXiv:1511.08060, 2015. https://doi.org/10.48550/arXiv.1511.08060
16. Deng J, Dong W, Socher R, Li L, Li K, Li FF (2009) ImageNet: a large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition; ; Vol. 2009, pp 248–255. https://doi.org/10.1109/CVPR.2009.5206848
17. Hunter JD (2007) Matplotlib: a 2D graphics environment. Comput Sci Eng 9:90–95. https://doi.org/10.1109/MCSE.2007.55
18. Zakaria N, Mohamed F, Abdelghani R, Sundaraj K (2021) VGG16, ResNet–50, and GoogLeNet deep learning architecture for breathing sound classification: a comparative study. In Proceedings of the 2021 International Conference on Artificial Intelligence for Cyber Security Systems and Privacy; ; Vol. 1, pp 1–6. https://doi.org/10.1109/AI-CSP52968.2021.9671124
19. Gong Y, Liu G, Xue Y, Li R, Meng L (2023) A survey on dataset quality in machine learning. Inf Softw Technol 162:107268. https://doi.org/10.1016/j.infsof.2023.107268
20. Mohanty SP, Hughes DP, Salathé M (2016) Using deep learning for image-based plant disease detection. Front Plant Sci 7:1419. https://doi.org/10.3389/fpls.2016.01419
21. Google Brain Team (2025) TensorFlow: An End-to-End Open Source Platform for Machine Learning; Google Brain Team, https://www.tensorflow.org/
22. NumPy Steering Council. NumPy: The Fundamental Package for Scientific Computing with Python; NumPy Steering Council (2025) https://numpy.org/
23. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Müller A, Nothman J, Louppe G, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay É (2012) Scikit-learn: machine learning in Python. arXiv preprint arXiv:1201.0490. https://doi.org/10.48550/arXiv.1201.0490
24. Roeder L, Netron (2026) Visualizer for Neural Network, Deep Learning, and Machine Learning Models. GitHub repository, https://github.com/lutzroeder/netron

# Figures




**Figure 1**

The architecture for transfer learning with VGG16 for plant disease classification. A sequential diagram of the model's architecture, which includes a rescaling layer, a VGG16 base model (unfrozen and fine-

tuned), an average pooling layer to increase efficiency, and an output  layer with Softmax activation that makes the model predictions. The architecture diagram was  generated using Netron[24] from a .keras file.

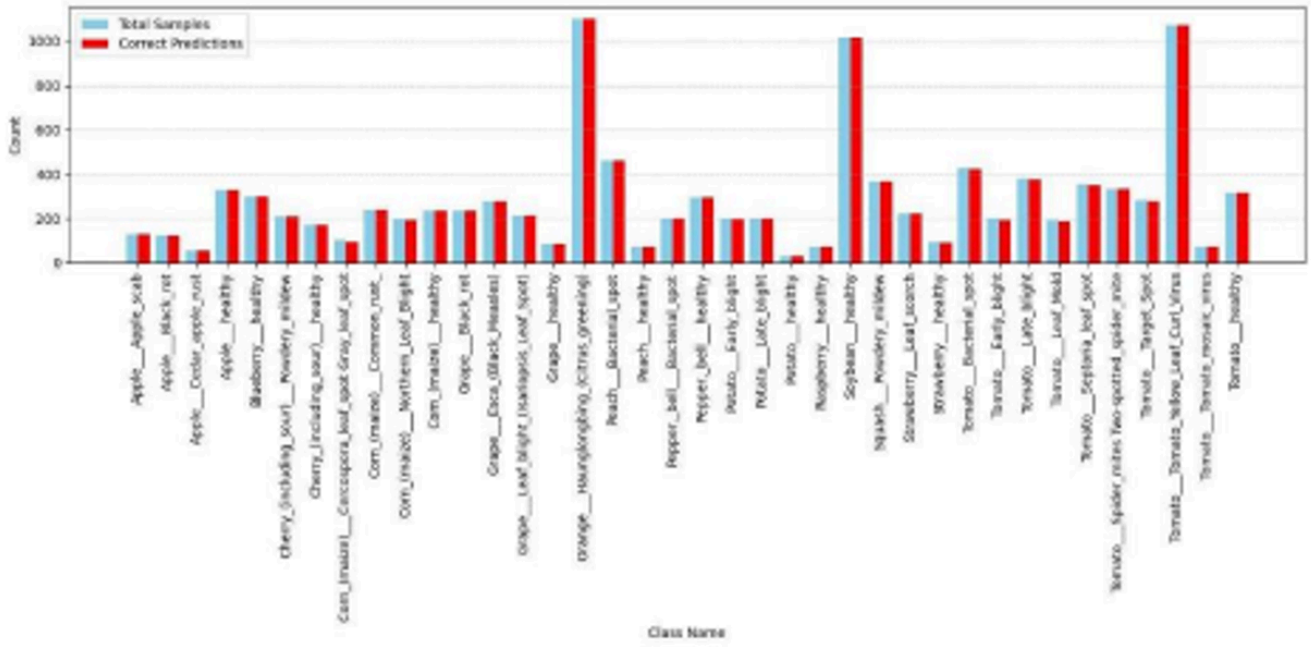



**Figure 2**

VGG16 model accuracy breakdown by class. The blue bars represent total samples in any given specific class of plant disease, while the red bars represent the total samples in that class that the model predicted correctly. Computed with Matplotlib, a Python library for graph drawing.

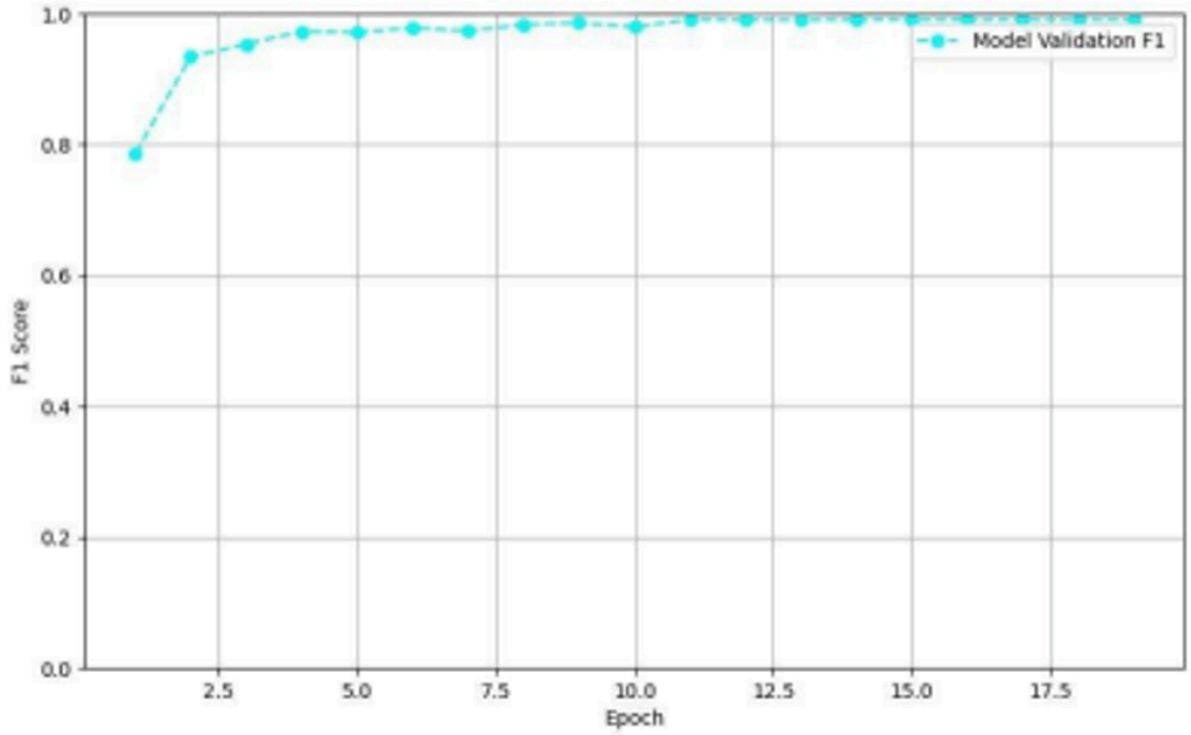



**Figure 3**

F1 score graph of the VGG16 model on the validation set during training. On the x-axis is the number of training epochs, while on the y-axis the corresponding F1 score is shown. Computed with Matplotlib.

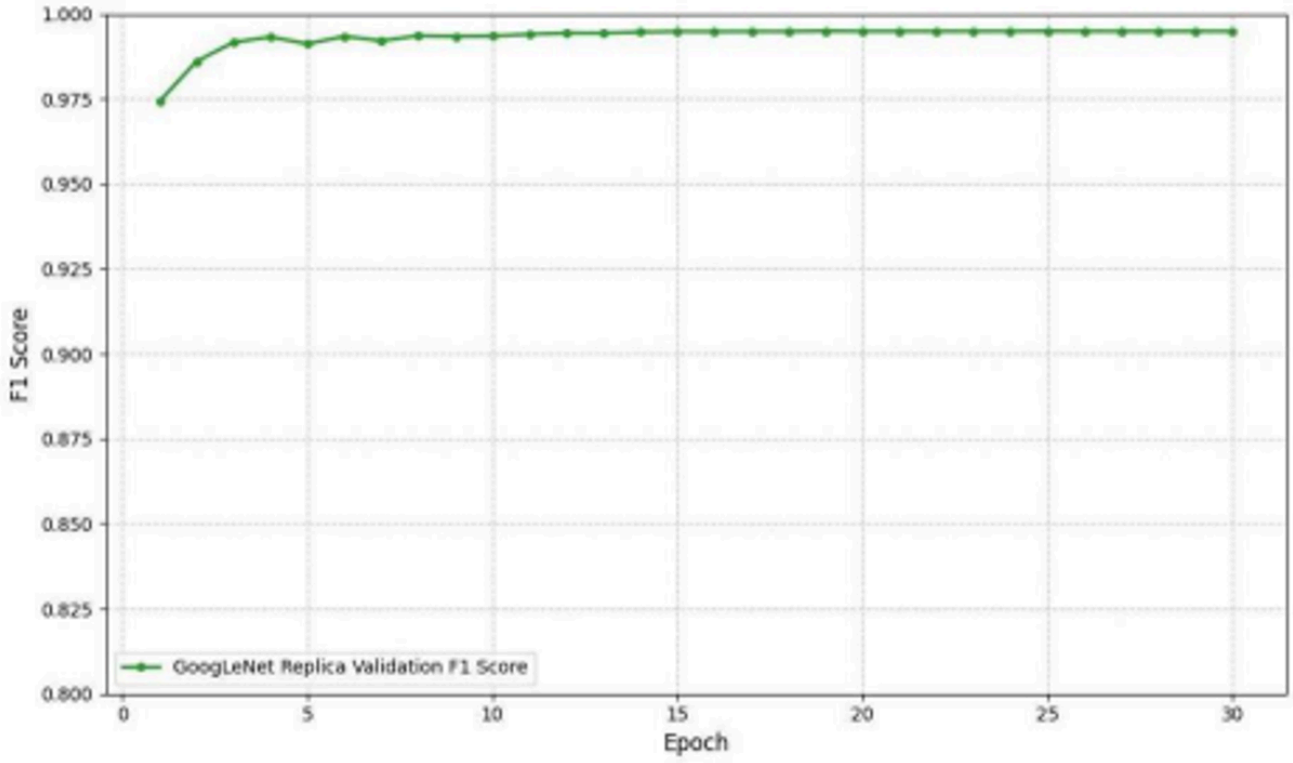


**Figure 4**

F1 score graph of the GoogLeNet model on the validation set during training**.** On the x-axis is the number of training epochs, while on the y-axis the corresponding F1 score is shown. Computed with Matplotlib.