

# Personal Assistant Development by CED (Canine Eye-disease Detection)

**K. Chun**

Department of Semiconductor Electronic Engineering, Daegu Catholic University,  
Gyeongsan, Gyeongbuk, Korea

---

## Article Info

### Article history:

Received Oct 21, 2023

Revised Dec 19, 2023

Accepted Dec 30, 2023

---

### Keyword:

CED

(Canine Eye-disease Detection),

Deep Learning,

MobileNet,

SqueezeNet,

---

## ABSTRACT

In this paper, we develop a deep learning-based canine eye disease detection and utilize it to create a dog health management system. With the recent surge in the number of pet dogs, ensuring their well-being has become crucial. We achieve this by applying lightweight deep learning methods like MobileNet and SqueezeNet to mobile devices, enabling regular monitoring of a pet's eye health. Additionally, we provide a GPS-based search feature for nearby hospitals, facilitating swift response to diseases. The validity of the developed method is demonstrated through experiments on 5 eye diseases. The results confirm the importance of considering appropriate recognition rates and recognizability metrics, as outcomes may vary depending on the applied deep learning approach.

*Copyright © 2023 Institute of Advanced Engineering and Science.  
All rights reserved.*

---

### Corresponding Author:

K. Chun,

Department of Semiconductor Electronic Engineering,

Daegu Catholic University,

Gyeongsan, Gyeongbuk, Korea

Email: [kchun@cu.ac.kr](mailto:kchun@cu.ac.kr)

---

## 1. INTRODUCTION

Artificial Intelligence (AI) has a well-documented history of displaying discriminatory trends within the context of human diseases [1-8]. However, the progress of AI in the veterinary domain has been impeded due to limitations in veterinary image datasets [9]. Obtaining consent from animal patients poses challenges, and veterinarians often refrain from capturing images during visits due to the hectic hospital environment. Despite the scarcity of accessible data, there exists a substantial demand for deep-learning (DL) applications among both veterinarians and pet owners, offering potential benefits across various aspects [10].

In light of these challenges, DL applications have the potential to enhance the accuracy and efficiency of diagnosis, ultimately leading to timely interventions and improved outcomes for animal patients. This study specifically focuses on achieving application in daily life, aiming to accurately diagnose diseases using casual images obtained by pet owners, despite the majority of training images being captured in hospitals.

There has been a growing focus on the exploration of DL algorithms applied to the classification of eye-related diseases through image analysis. For instance, Junayed et al. introduced CataractNet, a deep neural network designed for the automatic detection of cataracts in fundus images [11]. Their proposed network demonstrates superior performance compared to existing methodologies, achieving an average accuracy of 99.13% while simultaneously reducing computational cost and runtime. Li et al. presented a DL system aimed at classifying keratitis, various corneal abnormalities, and normal corneas based on slit-lamp images [12]. Christopher identified glaucomatous damage in optic nerve head (ONH) fundus images [13]. Aranha's study encompassed cataracts, diabetic retinopathy, excavation, and blood vessels; however, individual binary classification networks were utilized to differentiate between normal and abnormal images

[14]. The majority of research in the field of ophthalmological image analysis has been confined to a select subset of diseases such as cataracts, corneal diseases, and glaucoma [11]. In contrast, our approach utilizes a unified model capable of identifying multiple diseases manifesting in distinct anatomical locations and applies to the mobile devices for the purpose of personal assistant

In deep learning, both MobileNet and SqueezeNet have been specifically devised to address the challenge of deploying deep learning models on mobile and embedded devices characterized by restricted computational resources. MobileNet achieves this by implementing depth-wise separable convolutions, thereby reducing both the model's size and computational demands. Conversely, SqueezeNet attains a compact model size through a judicious combination of 1x1 and 3x3 convolutions. While MobileNet generally exhibits superior accuracy compared to SqueezeNet, the latter effectively strikes a balance between model size and accuracy. In scenarios where the prioritization lies in minimizing model size and computational resources, SqueezeNet emerges as the apt choice, particularly for applications marked by constrained memory and processing capacities.

In this paper, we apply a deep learning-based electronic dog eye disease diagnostic technology and develop a pet health management system using it. Recently, deep learning has been applied in various fields, and health diagnosis is one of the major application areas. Especially, with the rapid increase in the number of pet dogs, managing their health has become crucial. However, it is challenging for non-experts to do so. While visiting an animal hospital is an option, it may be difficult to determine when an illness has occurred, and if there is no hospital nearby, there may be no basis for judgment. By utilizing lightweight deep learning methods such as MobileNet or SqueezeNet, we make it possible to apply the technology to mobile devices. This allows individuals to regularly check their pet's eye health, and if a problem is detected, they can be promptly guided to a nearby hospital for necessary measures against diseases, enabling them to maintain their health. The validity of the developed method was demonstrated through experiments on 5 eye diseases. The results confirmed the difference in sample size and recognizability according to the deep learning method, considering the recognition success rate and establishing an appropriate benchmark model.

## 2. DEEP LEARNING FOR MOBILE DEVICES AND PERSONAL ASSISTANT

### 2.1. Deep Learning for Mobile Devices

#### CNN

The network architecture design refers to the fundamental framework of the network, which delineates the quantity of units and the interconnections among distinct groups of units, typically referred to as layers.

The architecture of a Convolutional Neural Network (CNN) encompasses the arrangement of convolutional, activation, pooling, and additional layers within a unified structure. It prescribes the overarching arrangement of the complete CNN, addressing inquiries pertaining to the total number of layers, the specific unit quantities within each layer, and the manner in which these units are interconnected.

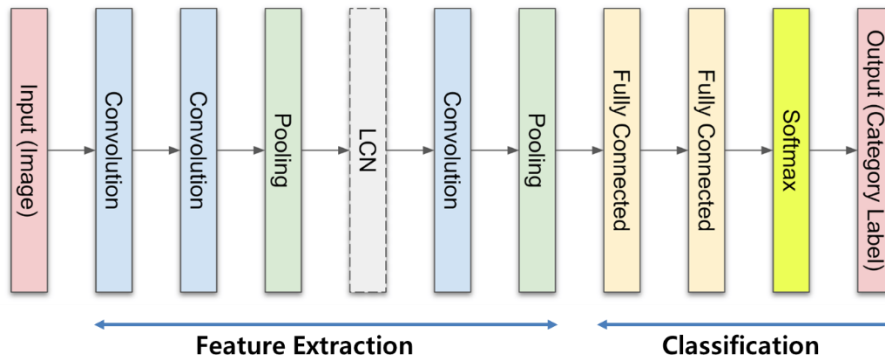


Figure 1. CNN basic structure

In a CNN, the model is structured through the repetition of convolution layers and pooling layers. The iterative methodology can vary depending on the design; for instance, multiple layers of convolution may be arranged, followed by a single pooling layer at the end. Additionally, the inclusion or exclusion of the Local Contrast Normalization (LCN) layer can be determined based on necessity. And the final classification results are computed using the softmax function. The softmax function is primarily utilized for categorizing more than two categories, employing probability values calculated for each category to perform classification.

### MobileNet and SqueezeNet

MobileNet and SqueezeNet are two prominent convolutional neural network (CNN) architectures designed specifically for efficient and lightweight deep learning on devices with limited computational resources. Despite sharing the common objective of minimizing computational and memory demands compared to conventional CNNs, they employ distinct methodologies to achieve this aim. This comparative analysis delves into the principal characteristics, architectural disparities, performance metrics, and practical applications of MobileNet and SqueezeNet.

Introduced by Howard et al. in 2017 [15], MobileNet places a strong emphasis on model efficiency while maintaining accuracy. It accomplishes this feat by employing depth-wise separable convolutions, which represent a factorized variant of conventional convolutions. Traditional convolutions entail the application of a singular filter across the entire input volume, resulting in a substantial computational overhead. In contrast, depth-wise separable convolutions partition the convolution process into two discrete operations: depth-wise convolutions and point-wise convolutions. Depth-wise convolutions individually apply a single filter to each input channel, effectively diminishing the computational complexity. Subsequently, point-wise convolutions amalgamate the outputs of depth-wise convolutions through  $1 \times 1$  convolutions, facilitating the capture of cross-channel interactions. By disentangling spatial and channel-wise filtering, MobileNet significantly curtails the number of parameters and computations, rendering it particularly suitable for deployment on mobile and embedded devices.

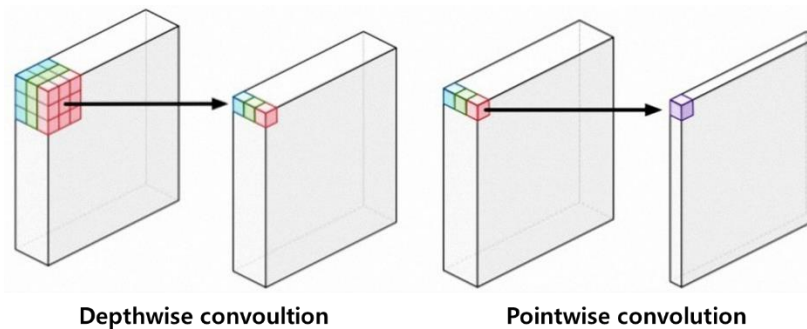


Figure 2. MobileNet

SqueezeNet, introduced by Iandola et al. in 2016[16], addresses the imperative of model efficiency through the incorporation of fire modules and the application of aggressive compression methodologies. Fire modules are composed of a squeeze layer employing  $1 \times 1$  convolutions, serving to diminish the quantity of input channels. This is succeeded by expand layers, which employ a combination of  $1 \times 1$  and  $3 \times 3$  convolutions to encapsulate spatial information. The squeeze layer functions as a bottleneck layer, effectively reducing computational complexity, while the expand layers facilitate the restoration of information that might have been lost. Additionally, SqueezeNet pioneers the "squeeze-excitation" paradigm to model interdependencies among channels. This approach leverages a concise set of global statistics to dynamically scale the output of each channel, thereby augmenting the representational capacity of the network.

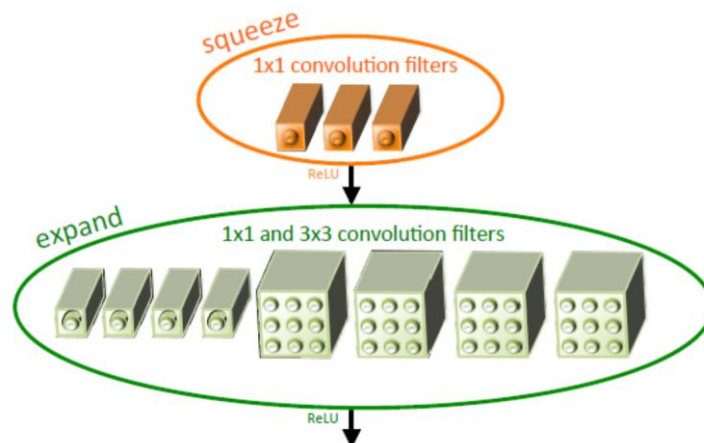


Figure 3. SqueezeNet

In architectural terms, MobileNet and SqueezeNet exhibit discernible distinctions. MobileNet encompasses multiple layers, including depth-wise separable convolutions, succeeded by global average pooling and a fully connected layer for classification. The architecture is amenable to customization through the adjustment of hyper parameters like depth multiplier and input resolution, allowing for a judicious balance between accuracy and efficiency. In contrast, SqueezeNet is characterized by fire modules interspersed with pooling layers and 1x1 convolutions, adhering to the conventional convolutional neural network (CNN) structure with convolutional layers, activation functions, and fully connected layers for classification.

The comparative evaluation of MobileNet and SqueezeNet encompasses various facets. MobileNet achieves commendable accuracy in image classification tasks, concurrently demonstrating substantial reductions in model size and computational complexity vis-à-vis conventional CNNs. This equilibrium between efficiency and accuracy renders it suitable for real-time applications on mobile and embedded platforms. Conversely, SqueezeNet attains accuracy levels akin to larger models, distinguished primarily by its significantly smaller footprint. It achieves notable compression ratios by leveraging aggressive compression techniques, including 1x1 convolutions and parameter sharing. Consequently, SqueezeNet excels in contexts where model size assumes paramount importance, particularly in deployment scenarios involving edge devices characterized by constrained memory and processing capabilities. In Figure 4, some results show the inference time of two methods [17] and SqueezeNet responses fast on constrained conditions (especially the device is old one).

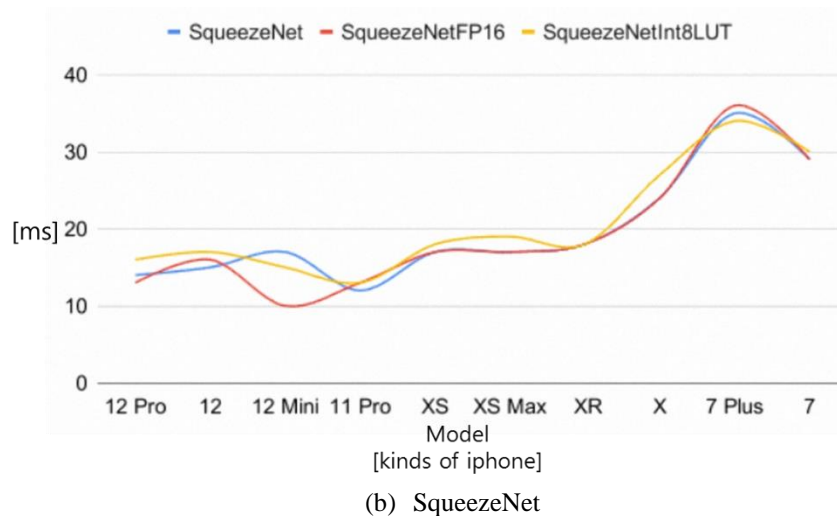
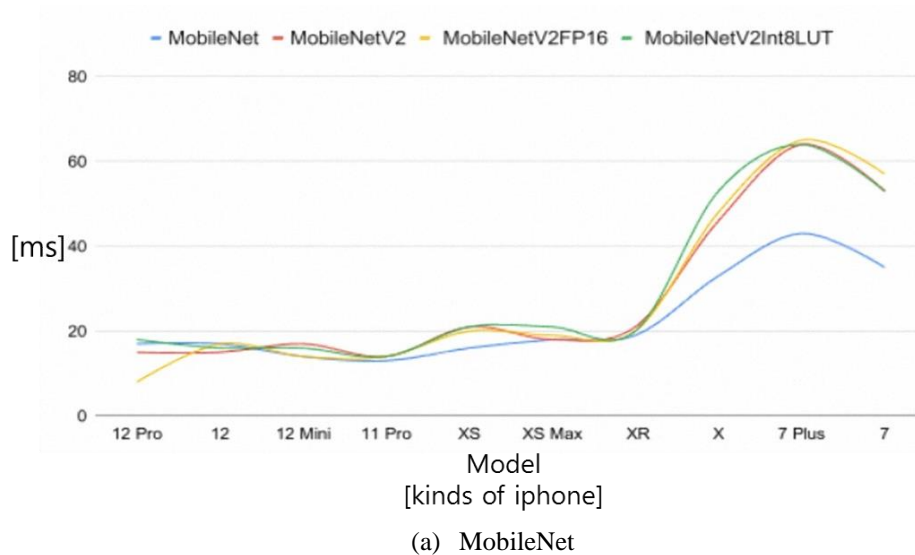


Figure 4. Total Inference time

The selection between MobileNet and SqueezeNet hinges on the precise exigencies of the application at hand. MobileNet, possessing adaptability and versatility, is an apt choice for a spectrum of resource constraints, while upholding a commendable level of accuracy. It finds relevance in tasks spanning object detection, semantic segmentation, and image classification on mobile platforms. Conversely, SqueezeNet, distinguished by its highly compressed model size, stands out in scenarios where storage and computational resources are severely constrained. It finds utility in low-power devices, applications within the Internet of Things (IoT) domain, and contexts where network bandwidth constitutes a limiting factor.

In summation, both MobileNet and SqueezeNet present proficient solutions for deep learning on resource-limited devices. MobileNet achieves efficiency through the employment of depth-wise separable convolutions, thereby diminishing both parameter count and computational load. In contrast, SqueezeNet capitalizes on fire modules and assertive compression techniques to substantially diminish model size, all the while upholding competitive levels of accuracy. The preference between the two architectures is contingent upon the specific requisites of the application, with MobileNet embodying versatility and adaptability, and SqueezeNet offering maximal compression for exceedingly constrained resources.

### **CED(Canine Eye-disease Detection)**

The models of MobileNet and SqueezeNet are as follows. First, the used layers are Convolution, Flatten, Fully Connected, and Fully Connected.

Table 1. Model

Layer	Structure
Convolution	5x5x1: 7x7x256→3x3x5
Flatten	3x3x5→45
Fully Connected	45→100
Fully Connected	100→ Number of Labels

The first layer describes a specific convolutional layer in the MobileNet architecture. It applies a 5x5 convolution to a single-channel input (such as a grayscale image), resulting in a 7x7 output with 256 channels. This output is then further processed by a 3x3 convolution with 5 channels. These operations are fundamental to how MobileNet processes information in its deep learning architecture. The "Flatten" layer is a common layer in many neural network architectures, including CNNs. Its purpose is to convert the multi-dimensional output of the previous layer into a one-dimensional array. Also known as a dense layer, the "Fully Connected" layer is a fundamental component in neural networks. In this layer, every neuron is connected to every neuron in the preceding and succeeding layers.

And the hyperparameters we used are specific settings that are crucial for training a machine learning model in MobileNet.

Table 2. Hyperparameters

Hyperparameters	Value
Learning Rate	0.0001
Epochs	20
Training Data Fraction	0.4
Optimizer	Adam

The learning rate determines the step size at which the model's parameters (weights) are updated during training. A smaller learning rate means smaller steps and slower convergence, but it can lead to more accurate results. A larger learning rate might speed up training but could overshoot the optimal values. An epoch is one complete pass through the entire training dataset. Training a model involves multiple epochs. During each epoch, the model's parameters are updated based on the loss calculated on the training data. More epochs generally allow the model to learn more, but too many epochs can lead to overfitting. The training data fraction indicates what fraction of the entire dataset is used for training. In this case, 40% of the available data is used for training, which means the rest (60%) is likely reserved for validation and testing. Adam (short for Adaptive Moment Estimation) is an optimization algorithm used for training deep learning models. It adapts the learning rates of individual parameters based on their past gradients. It's a popular choice because it generally works well for a wide range of problems.

## 2.2. Personal Assistant

The development of an app utilizing AI to recognize and examine a dog's eyes represents an effort to leverage technology for canine health improvement. While this app may not be an exceedingly specialized project, it serves the purpose of exploring the potential of AI in the field of dog eye recognition.

The research results demonstrate a reasonable level of accuracy in recognizing and analyzing dog eyes for potential ailments. However, it is important to acknowledge the limitations of this app, which is not highly specialized, and therefore may exhibit expectedly lower accuracy and restricted functionality. As a result, the primary objective of this app is to aid in the early assessment of a dog's health rather than providing definitive diagnoses. Therefore, for precise diagnosis and treatment, it is advised to visit a nearby veterinary clinic if a disease is suspected.

User feedback and performance evaluations of the app play a crucial role in future enhancements. To improve the app's accuracy, additional refinement of the AI algorithms and the acquisition of more canine eye data will be necessary.

Overall, this research demonstrates the potential of artificial intelligence in the field of dog eye recognition. With further development, improvement, and integration driven by user feedback, this app has the potential to positively impact canine well-being, providing valuable support for both dog owners and veterinarians in monitoring and enhancing eye health.

The program is structured with three main screens. The initial screen serves as a cover page with menu selection options. Users can perform a preliminary diagnosis of their dog's condition through the "Eye health test" menu and easily proceed with the steps for hospital visits through the "Find nearby hospitals" menu.

The cover page is organized as follows: the "Eye health test" menu is located on the left, while the "Find nearby hospitals" menu is positioned on the right in Figure 5.



Figure 5. Cover

The "Eye health test" menu allows users to begin by touching the magnifying glass icon to capture a photo. Additionally, users have the option to choose between the front and rear cameras if needed. Once the photo is taken, the algorithm utilizes provided experimental data along with existing reference data to conduct a preliminary diagnosis of potential ailments. It then displays the three most likely scenarios, starting from the left, as seen in Figure 6. Particularly for the most probable scenario, an explanation is provided below for easy comprehension of the potential ailment. Users are recommended to verify this information and, in cases of high probability, to visit a nearby animal hospital for further diagnosis and treatment.



Figure 6. Camera diagnosis by CED

The "Find nearby hospitals" menu offers additional functionalities and is primarily designed for services within Korea. As a result, it is structured around a map search displayed in Korean. (Future improvements will aim to enable English and multilingual services.) Users can search for animal hospitals using place names, and the results are presented based on proximity using GPS. This result can also be displayed based on keyword-related accuracy. The menu provides information such as the hospital's address, phone number, directions, and navigation. Additionally, it offers a "Like" count, allowing users to consider it as a criterion when choosing which animal hospital to visit. This criterion is determined based on user feedback and proves to be a helpful aid in searching for animal hospitals.





Figure 7. Find nearby hospitals

### 3. TEST AND VALIDATION

To conduct the dog eye health examination, we first gathered a diverse set of data. In order to ensure the reliability of the information, we primarily utilized data from animal hospitals. However, high-quality and consistent data were not available due to various reasons, we did the acquisition of data utilizing smartphone cameras. This data source represents information attainable in real-life scenarios and aligns with the objective of this paper: to provide a guide for pet owners in real-life scenarios regarding the diagnosis of their pets' diseases.

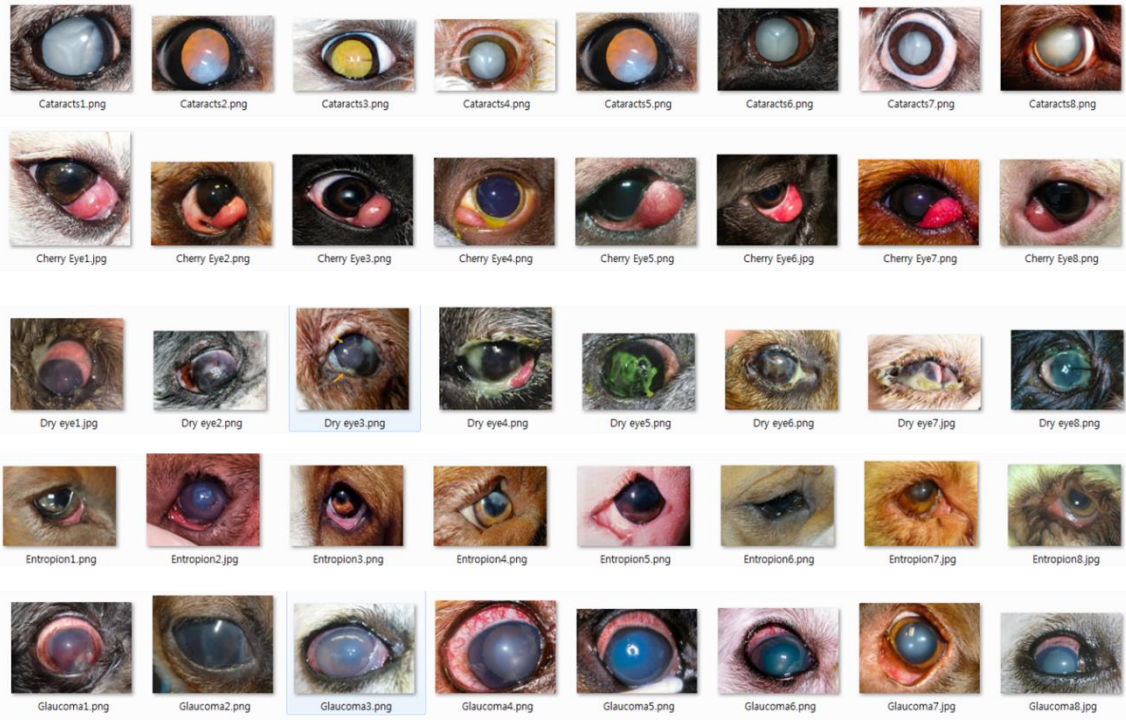
At first, we can collect data on 15 different types of eye diseases and also gathered data on the eyes of healthy dogs, but the numbers of samples are not enough for all diseases. For the experiment, it is essential to have an equal number of samples for each disease. Therefore, five diseases with an equal number of samples are selected for experimentation from Table 3.

Table 3. Dog eye disease

Collected eye diseases:	Cataracts, Cherry eye, Dry eye, Entropion, Glaucoma
-------------------------	---



Therefore, experiments were conducted on five diseases with an equal number of applicable instances: Cataracts, Cherry eye, Dry eye, Glaucoma, and Entropion. The samples used for the experiment, as illustrated in Figure 8, totaled 60 in number. Among these, 40 samples were allocated, with each of the five diseases represented by 8 unique instances, without any instances being duplicated. Conversely, there were 20 instances of healthy samples, constituting half of the total eye disease sample set.



(a) Sick



(b) healthy

Figure 8. Dog eyes

To conduct an eye health examination using deep learning, it is necessary to first create a model. Since the performance of deep learning is influenced by the amount of benchmark data. In this paper, experiments were conducted in two groups based on the number of samples: a group with a small number of samples and another group with twice as many samples. Firstly, in the group with a small number of samples, tests were performed by altering the quantity of utilized samples. In this context, an equal number of samples were applied for both diseased and healthy cases while varying the total sample count. Subsequently, in the case of using double the number of samples, the quantity of healthy samples remained constant while conducting experiments, which were then compared with those from the group with a smaller sample size.

In scenarios with a smaller sample count, as depicted in Table 4, 2, 3, and 4 samples were utilized for each disease, and an equivalent number of healthy samples were employed to differentiate between healthy and diseased cases. Consequently, for 10 disease sample cases, 10 healthy samples were utilized; for

15 disease sample cases, 15 healthy samples were employed; and for 20 disease sample cases, 20 healthy samples were utilized during the model training.

Table 4. Number of samples (small samples)

Model (number of samples)	Sick	Healthy
Smodel10	2 photos x 5 diseases = 10 samples	10 samples
Smodel15	3 photos x 5 diseases = 15 samples	15 samples
Smodel20	4 photos x 5 diseases = 20 samples	20 samples

Using this, we constructed three benchmark models based on the number of samples and conducted tests for nine different cases. The experimental results are shown in Figure 9. We considered results with an accuracy of 70% or higher as successful and highlighted them in yellow. In the tests, we displayed the top three cases for each cell with the highest recognition rates in order. While it is common to consider the one with the highest probability, we implemented the feature to display up to the third place with the intention of providing guidance for other possible diseases.

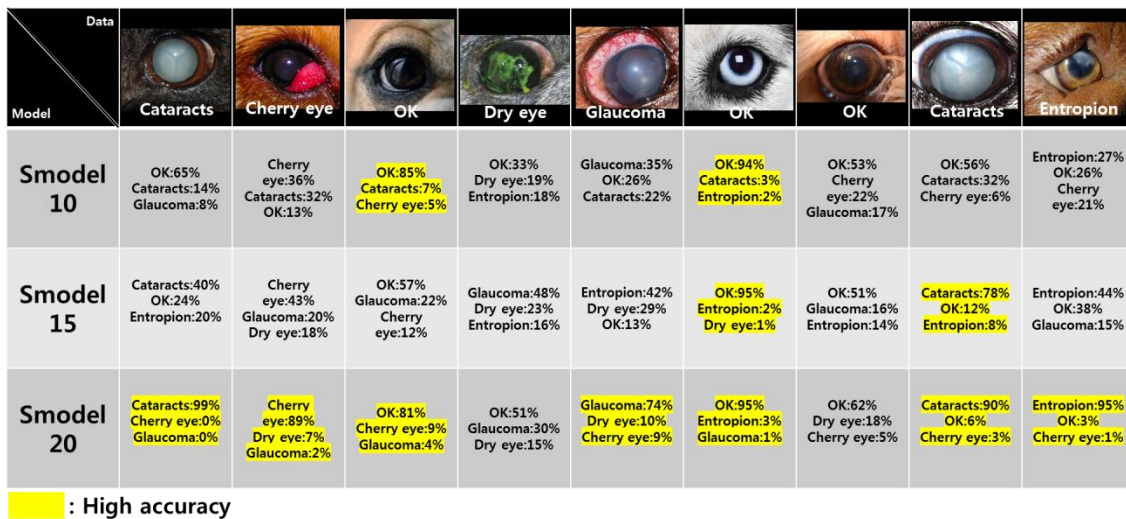


Figure 9. Results according to number of samples (small samples)

Firstly, in the case of 10 samples(Smodel10), meaning each disease used in the model has 2 samples, as seen in the figure, the relatively higher number of healthy cases (10 samples) shows positive recognition results. Conversely, the recognition results for the other diseases are relatively low. In the case of 15 samples (3 samples for each disease,Smodel15), similar results can be observed. However, unlike the case with 10 samples where only healthy cases were recognized, in this case, there is one case each of healthy and diseased cases, but the recognition success rate remains the same at 22% for all test cases. This is still a challenging figure for practical application.

In the case of 20 samples(Smodel20), where 4 samples were used for each disease, the recognition success rate starts to show meaningful results, reaching around 78%. This indicates that using four or more samples enables meaningful recognition, and the actual recognition potential can go up to 99%. However, in this scenario, a significantly larger number of healthy samples, five times more, should be used compared to the number of samples representing the diseased cases.

The confusion matrix of Smodel20 is in Table 5 and the accuracy is as follows:

$$Accuracy=(70+72+71+70+72+91)/596=0.748$$

Table 5. Confusion matrix(Smodel20)

	Cataracts	Cherry eye	Dry eye	Entropion	Glaucoma	OK
Cataracts	70	0	4	0	3	0
Cherry eye	0	72	5	10	10	0
Dry eye	5	8	71	0	30	15
Entropion	0	5	0	70	0	10
Glaucoma	5	9	20	0	72	5
Ok	0	0	0	5	1	91

Next, we compared the application results of MobileNet and SqueezeNet. The sample size was fixed at 20, and for each case, we constructed a baseline model. The experimental results for recognition according to each model are shown in Figure 10.

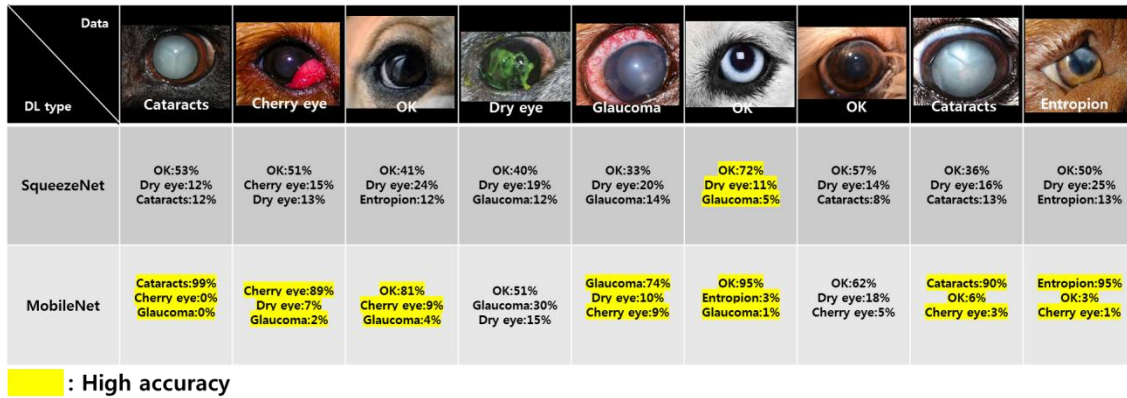


Figure 10. MobileNet vs. SqueezeNet (small samples)

Looking at the recognition success rates, SqueezeNet shows a significantly lower rate of 11%(Recognize one of 9 tests) compared to MobileNet's 78%(Recognize 7 of 9 tests). As explained in the previous section, this can be attributed to the performance degradation characteristic of an approach that emphasizes implementation of deep learning in limited and light weight scenarios.




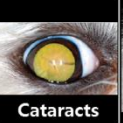




In the case of a larger sample size, as presented in Table 6, for each disease, twice the number of samples compared to the smaller sample group were used, specifically 4, 6, and 8 samples. However, the number of healthy samples remained fixed at 12, a number higher than the relative number of healthy samples for each disease compared to the diseased samples in the smaller sample test. This decision was made to ensure that the fixed number of healthy samples was relatively higher than the sample count for each disease, considering that in the smaller sample test, there were relatively more healthy samples per disease. Therefore, for 20, 30, and 40 disease samples, 12 healthy samples were consistently utilized during the model training.

Table 6. Number of samples (double samples)

Model (number of samples)	Sick	Healthy
Dmodel20	4 photos x 5 diseases = 20 samples	12 samples
Dmodel30	6 photos x 5 diseases = 30 samples	12 samples
Dmodel40	8 photos x 5 diseases = 40 samples	12 samples

Using this, we constructed three benchmark models based on the number of samples again and conducted tests for eight different cases. The experimental results are shown in Figure 11. As before, we considered results with an accuracy of 70% or higher as successful and highlighted them in yellow. In the tests, we displayed the top three cases for each cell with the highest recognition rates in order.



Data								
Model	Glaucoma	Cherry eye	Entropion	Cataracts	OK	Cataracts	Dry eye	Cherry eye
<b>Dmodel 20</b>	Glaucoma:33% OK:18% Cataracts:16%	Cherry Eye:83% OK:13% Cataracts:2%	OK:96% Entropion:3% Cherry Eye:0%	Cherry Eye:25% Cataracts:21% OK:20%	OK:92% Cataracts:4% Cherry Eye:2%	Cataracts:37% Cherry Eye:33% OK:10%	OK:47% Dry Eye:29% Cherry Eye:19%	OK:87% Dry Eye:8% Glaucoma:2%
<b>Dmodel 30</b>	OK:18% Cataracts:17% Entropion:17%	Cherry Eye:77% Entropion:12% Dry Eye:7%	Entropion:67% Dry Eye:10% Cataracts:8%	Dry Eye:21% Glaucoma:18% Cataracts:17%	OK:90% Dry Eye:3% Cherry Eye:2%	Cataracts:44% Cherry Eye:22% OK:12%	Dry Eye:37% Entropion:20% Cherry Eye:15%	Cherry Eye:63% Entropion:25% Dry Eye:7%
<b>Dmodel 40</b>	Glaucoma:62% Entropion:10% Cherry Eye:10%	Cherry Eye:70% Entropion:20% OK:5%	Entropion:75% Cherry Eye:19% OK:5%	Cataracts:74% OK:10% Glaucoma:10%	OK:91% Entropion:5% Cherry Eye:2%	Cataracts:53% Glaucoma:42% Dry Eye:4%	Dry Eye:79% Glaucoma:8% Cataracts:6%	Cherry Eye:74% Entropion:13% Cataracts:7%


 : High accuracy

Figure 11. Results according to number of samples (double samples)

In the case of 20 samples (Dmodel20), despite using an identical set of 20 disease samples, only two instances of success are demonstrated. This alteration stems from the reduction of healthy samples from the previous 20 to 12, leading to an error in recognizing the disease samples as healthy eyes in two out of three attempts, as depicted in Figure 9. Similarly, in the scenario of 30 samples (Dmodel30), doubling the disease sample count also results in only two instances of successful recognition, mirroring the outcomes of Figure 9. These outcomes pose considerable challenges for practical implementation.

However, with 40 samples (Dmodel40), employing eight samples for each disease, the recognition success rate notably improves, reaching approximately 75%. This signifies that when the number of disease samples and healthy samples reaches an optimal level (around 10 each), a model using an equivalent number of samples can sufficiently provide guidelines for pet owners in everyday life for dog eye disease assessment.

Even in scenarios where the disease sample count is low but there is a surplus of healthy samples (Smodel20), recognition becomes possible. However, when the number of disease and healthy samples is similar, Dmodel40 emerges as a practical model, demonstrating the potential for actual recognition capabilities of up to 91%.

The confusion matrix of Smodel20 is in Table 7 and the accuracy is as follows:

$$\text{Accuracy}=(75+76+75+75+74+80)/584=0.779$$

Table 7. Confusion matrix(Dmodel40)

	Cataracts	Cherry eye	Dry eye	Entropion	Glaucoma	OK
Cataracts	75	0	4	5	10	10
Cherry eye	0	76	4	20	1	5
Dry eye	6	0	75	4	8	5
Entropion	0	10	0	75	0	5
Glaucoma	0	9	5	10	74	0
OK	0	2	0	5	1	80

The accuracy, as computed using the confusion matrix, resulted in 74.8% for Smodel20 and 77.9% for Dmodel40. Comparatively, this demonstrates a relative 8% decrease in accuracy compared to another study (84.7% [18]), which is chosen for comparison with the perspective of developing a practical deep learning framework for classifying ocular surface disease images in companion animals. However, this discrepancy 8% could be attributed to the disparity between the paper's[18] utilization of data acquired from specific equipment in veterinary clinics, whereas the present study is based on casual images.

At last, we compared the application results of MobileNet and SqueezeNet for double sample(Dmodel40). The sample size was fixed at 40, and for each case, we constructed a baseline model. The experimental results for recognition according to each model are shown in Figure 12.

Data								
DL type	Glaucoma	Cherry eye	Entropion	Cataracts	OK	Cataracts	Dry eye	Cherry eye
SqueezeNet	OK:18% Cherry eye:18% Dry eye:18%	Cherry eye:17% Glaucoma:17% Dry eye:17%	OK:17% Cataracts:17% Dry eye:17%	OK:17% Cataracts:17% Dry eye:17%	OK:21% Cataracts:20% Dry eye:18%	Cherry eye:17% Glaucoma:17% Dry eye:17%	Cherry eye:17% Glaucoma:17% Dry eye:17%	Cherry eye:17% Glaucoma:17% Dry eye:17%
MobileNet	Glaucoma:62% Entropion:10% Cherry Eye:10%	Cherry Eye:70% Entropion:20% OK:5%	Entropion:75% Cherry Eye:19% OK:5%	Cataracts:74% OK:10% Glaucoma:10%	OK:91% Entropion:5% Cherry Eye:2%	Cataracts:53% Glaucoma:42% Dry Eye:4%	Dry Eye:79% Glaucoma:8% Cataracts:6%	Cherry Eye:74% Entropion:13% Cataracts:7%

: High accuracy

Figure 12. MobileNet vs. SqueezeNet (double samples)

In terms of recognition success rates, SqueezeNet demonstrates a significant inability to recognize as opposed to MobileNet's 75% (identifying 6 out of 8 samples). As observed in Smodel and confirmed in Dmodel, MobileNet appears to be a viable deep learning method in the current experimental scenario. The experiments conducted using Smodel and Dmodel highlighted the importance of appropriately considering the number of disease samples and healthy samples for effective training. Particularly noteworthy is the influence of the quantity of healthy eye samples on the assessment of eye diseases. This is attributed to the absence of specific patterns in healthy eyes, making it challenging to distinguish visual characteristics. Furthermore, given the diverse breeds of pets and variable lighting conditions in real-life settings, this study underscores the possibility of implementing disease recognition based on data acquired from real-life scenarios, utilizing DL methods on mobile devices, leveraging an appropriate number of samples, as opposed to traditional computer vision methods conducted in laboratory settings.

**Remark:** The mobile device used in the experiment is the Galaxy Z Flip3 model, running on the Android 13 operating system. It is equipped with the Qualcomm Snapdragon 888 SM8350 Platform and 8GB of LPDDR5 SDRAM. While response times were not explicitly measured during the actual tests, the recognition results appeared promptly, indicating no issues for practical use. Therefore, I did not feel the need to switch from MobileNet to SqueezeNet.

**4. CONCLUSION**

In this paper, we have developed a pre-screening system for assessing a dog's eye health using lightweight deep learning techniques, allowing for preliminary examinations before visiting an animal hospital. This system enables anyone, regardless of expertise, to perform diagnoses. We compared the application of two lightweight deep learning methods, MobileNet and SqueezeNet. The examination method developed using these lightweight deep learning techniques demonstrated performance levels suitable for practical use on mobile devices, particularly in terms of processing speed and constrained environments. Furthermore, we observed variations in performance results depending on the quantity of data used for the reference model, highlighting that even with lightweight methods, outcomes can vary. We anticipate that employing a larger set of reference data will enhance recognition capabilities, leading to improved recognition rates in the future. Also this research starts from collaborative research and development with the agricultural ICT company, SAMS and it is being pursued for a new startup venture. Thus the method used for CED can be expanded to apply to farm animals such as cattle and pigs, providing additional services for livestock management.

**ACKNOWLEDGEMENTS**

This work was supported by research grants from the Daegu Catholic University in 2023.

**REFERENCES**

[1] Ciompi F, Chung K, Riel S, Setio A, Gerke P, Jacobs C, Scholten E, Schaefer-Prokop C, Wille M, Marchiano A, et al. Towards automatic pulmonary nodule management in lung cancer screening with deep learning. arXiv preprint arXiv:1610.09157. 2016.

[2] Gulshan V, Peng L, Coram M, Stumpe M, Wu D, Narayanaswamy A, Venugopalan S, Widner K, Madams T, Cuadros J, et al. Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. *Journal of American Medical Association*, 2016; 316(22): 2402-2410.

- [3] Paul R, Hawkins S, Hall L, Goldgof D, Gillies R. Combining deep neural network and traditional image features to improve survival prediction accuracy for lung cancer patients from diagnostic CT. *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2016; 2570-2575.
- [4] Wimmer G, Hegenbart S, Vecsei A, Uhl A. Convolutional neural network architectures for the automated diagnosis of celiac disease. *International Workshop on Computer-Assisted and Robotic Endoscopy*, 2016; 104–113.
- [5] Chen J, Wu L, Zhang J, Zhang L, Gong D, Zhao Y, Hu S, Wang Y, Hu X, Zheng B, et al. Deep learning-based model for detecting 2019 novel coronavirus pneumonia on high-resolution computed tomography: a prospective study. *medRxiv*, 2020.
- [6] Purswani J. M., Dicker A. P., Champ C. E., Cantor M., Ohri N. Big data from small devices: The future of smartphones in oncology. *Seminars in radiation oncology*. 2019; 29(4): 338-347.
- [7] Lawanont W., Inoue M., Mongkolnam P., Nukoolkit C. Neck posture monitoring system based on image detection and smartphone sensors using the prolonged usage classification concept. *IEEJ Transactions on Electrical and Electronic Engineering*. 2018; 13(10): 1501-1510.
- [8] Razzak M. I., Naz S., Zaib A. Deep learning for medical image processing: Overview, challenges and the future. *Classification in BioApps*. Springer. 2018; 323-350.
- [9] Liu Y., Sun S. SagaNet: A small sample gated network for pediatric cancer diagnosis. *38th International Conference on Machine Learning*, 2021; 139: 6947–6956.
- [10] Arsomngern P., Numcharoenpinij N., Piriyyataravet J., Teerapan W., Hinthong W., Phunchongharn P. Computer-aided diagnosis for lung lesion in companion animals from X-ray images using deep learning techniques. *IEEE 10th International Conference on Awareness Science and Technology (iCAST)*, 2019; 1–6.
- [11] Junayed M. S., Islam M. B., Sadeghzadeh A., Rahman S. CataractNet: An automated cataract detection system using deep learning for fundus images. *IEEE Access*, 2021; 9: 128799–128808.
- [12] Li Z., Jiang J., Chen K., Chen Q., Zheng Q., Liu X., Weng H., Wu S., Chen W. Preventing corneal blindness caused by keratitis using artificial intelligence. *Nature Communications*, 2021; 12(1): 1–12.
- [13] Christopher M., Belghith A., Bowd C., Proudfoot J. A., Goldbaum M. H., Weinreb R. N., Girkin C. A., Liebmann J. M., Zangwill L. M. Performance of deep learning architectures and transfer learning for detecting glaucomatous optic neuropathy in fundus photographs. *Science Report*, 2018; 8(1): 1–13.
- [14] Aranha G. D. A., Fernandes R. A. S., Morales P. H. A. Deep transfer learning strategy to diagnose eye-related conditions and diseases: An approach based on low-quality fundus images. *IEEE Access*, 2023; 11: 37403–37411.
- [15] Howard A. G., Zhu M., Chen B., Kalenichenko D., Wang W., Weyand T., Andreetto M., Adam H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*. 2017.
- [16] Iandola F.N., Han S., Moskewicz M.W., Ashraf K., Dally W.J., Keutzer K. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and textless 1MB model size. *5<sup>th</sup> International Conference on Learning Representations*. 2016.
- [17] Tulasi A, Malarselvi S, Pappu M. A Review on MobileNet, ResNet and SqueezeNet for iOS & iPadOS for on Device Training and Prediction using CoreML. *International Research Journal of Engineering and Technology*. 2021; 08(06): 716-718.
- [18] Nam M. G., Dong S.Y. Classification of Companion Animals' Ocular Diseases: Domain Adversarial Learning for Imbalanced Data. *IEEE Access*. 2023; 11:143948-143955.