**DEEP LEARNING FOR AI**



**FINAL GROUP PROJECT REPORT**

**COMPETITION: UNIFESP X-ray Body Part Classifier Competition**

**TEAM8 MEMBERS**

**김진현**

**김유진**

**김희영**

**유현우**

**이호용**

**INDEX**

**1. Introduction of Competition & Data**

**2. EDA**

**3. Data Preprocessing**

**4. Modeling**

**4-0. Cross validation& Data augmentation**

**4-1. MODEL1: VGG 19**

**4-2. MODEL2: Efficient net B0**

**5. After participating this Competition**

**1. Introduction of Competition & Data**

We participated in UNIFESP X-ray Body Part Classifier Competition. The goal of this competition is to build an algorithm to correctly classify body parts in X-rays. And the deadline is August 1st, 2022.

텍스트, 실내, 창문, 바둑판식이(가) 표시된 사진

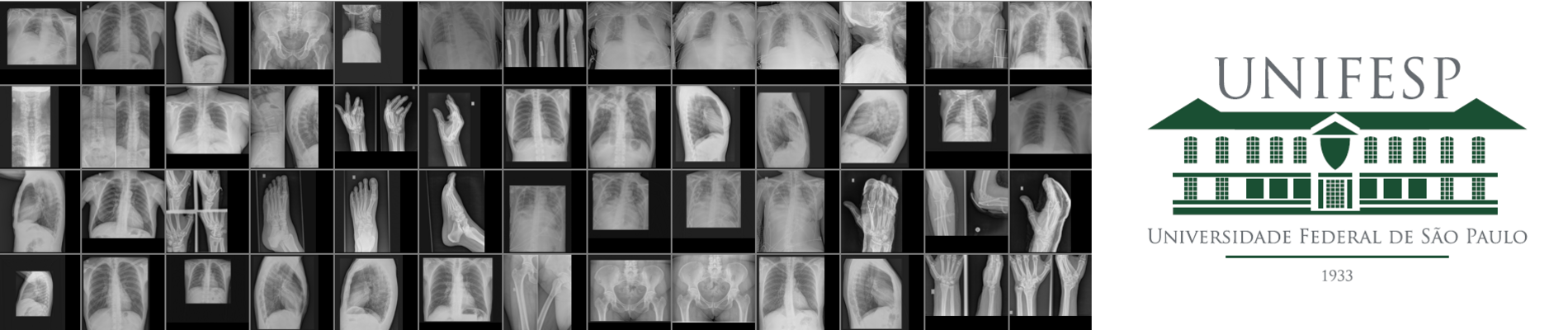
자동 생성된 설명

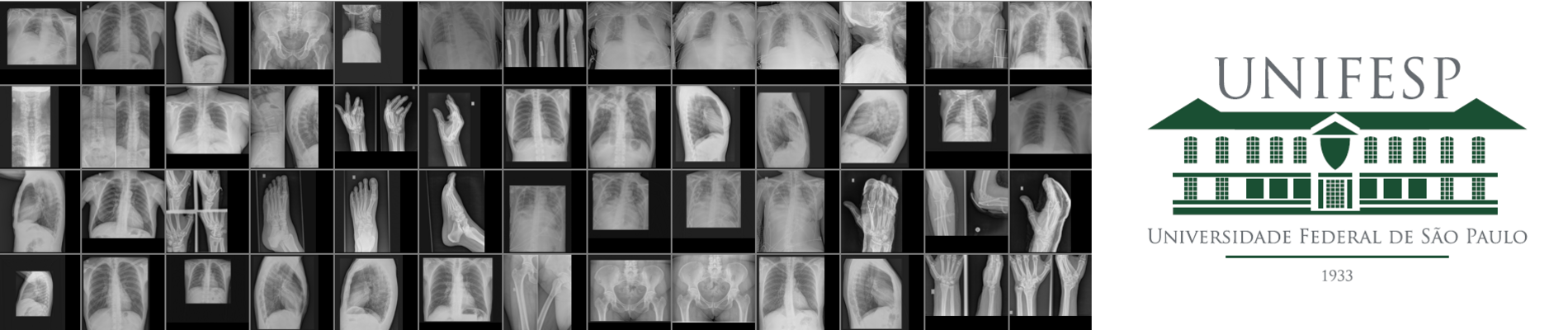
The X-ray images are labeled as at least one of 22 integers that each represents body parts.

As you can see below, abdomen is labeled as 0, ankle 1, cervical spine 2, and so on.

텍스트이(가) 표시된 사진

자동 생성된 설명





Here’s the data that was given by the competition.

텍스트이(가) 표시된 사진

자동 생성된 설명

First, this is train.csv, which contains 1738 data sets.

테이블이(가) 표시된 사진

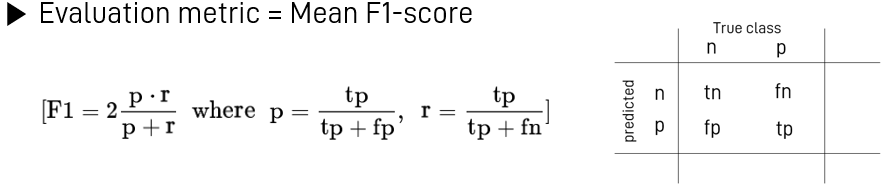
자동 생성된 설명

The first column is SOPInstanceUID, and this corresponds to a unique image, which is used in medical field. This is like the personal ID of human.

And the second column is the target values which represents the labels assigned to each sample.

This train.csv will be used for training.

There was train& test folder. These folders consist of DICOM images. DICOM stands for Digital Imaging and Communications in Medicine, which is also used is medical field. Train folder contains 1738 images, and the test folder contains 743 images. Test images will be used for the evaluation of the model that we built by comparing the prediction and the true label. Evaluation metric was mean f1-score.



The F1 score is defined as the harmonic mean of precision and recall.

The harmonic mean is an alternative metric for the more common arithmetic mean. It is often useful when computing an average rate.

In the F1 score, we compute the average of precision and recall. They are both rates, which makes it a logical choice to use the harmonic mean.

Since the F1 score is an average of Precision and Recall, it means that the F1 score gives equal weight to Precision and Recall:

-A model will obtain a high F1 score if both Precision and Recall are high

-A model will obtain a low F1 score if both Precision and Recall are low

-A model will obtain a medium F1 score if one of Precision and Recall is low and the other is high

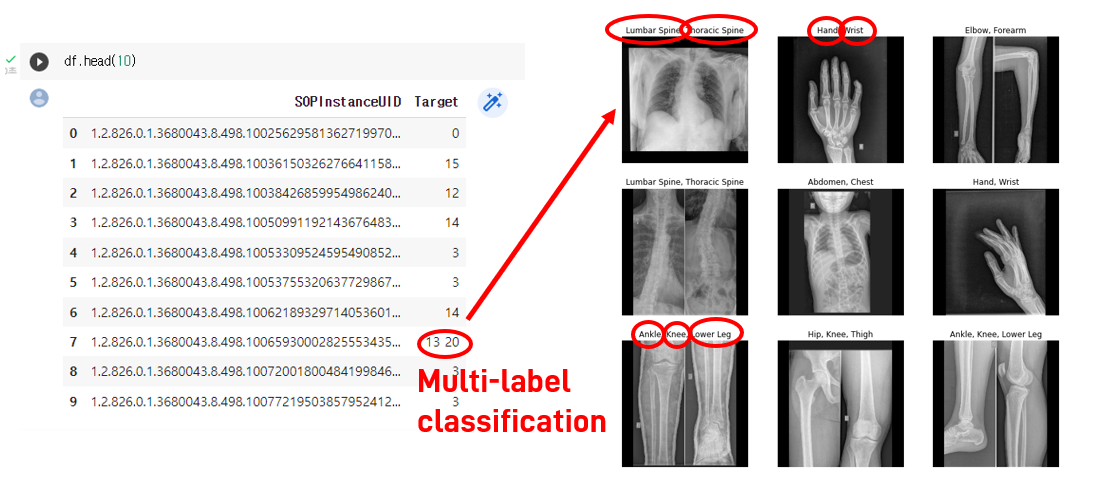
The reason we choose this competition.

Our task is one of medical image analysis. And there has been many promising advances in medical analysis tasks using deep learning techniques, such as automated cancer detection.

And CNN has been performing well in detecting many diseases like Alzheimer’s disease, Parkinson’s disease, and many other diseases. Also, there was an approach of detecting COVID-19 from chest X-Ray images using CNN. So, we thought this competition would be meaningful and worth participating as automated images diagnosis has brought and may bring many benefits to human health care system, like enhancing the efficiency in medical treatment.

**2. EDA**

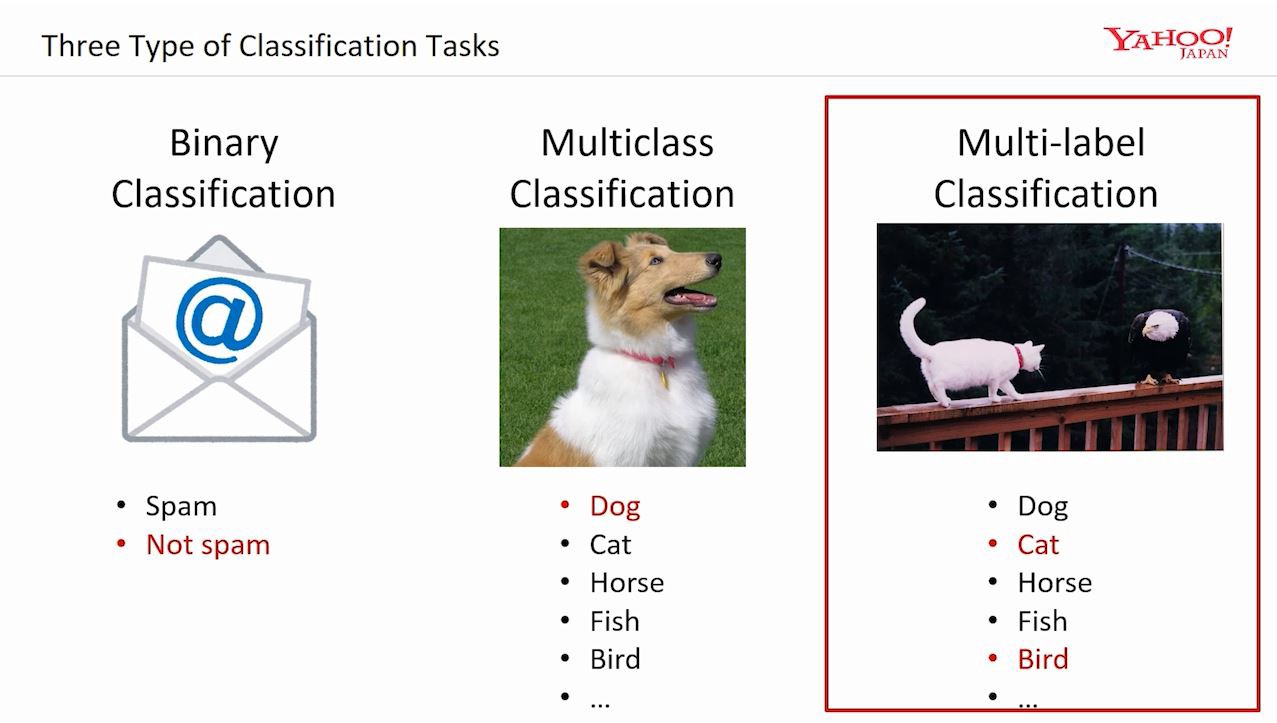
Each X-ray image is labeled as “at least” one of 22 integers.



There are some Targets that contains two or more labels like 8th data which is labeled 13 and 20 at once. These are the X-ray images that are assigned to two or more labels. The upper left image is the image of 8th data. This image is classified as ‘lumbar spine’ and ‘thoracic spine’.

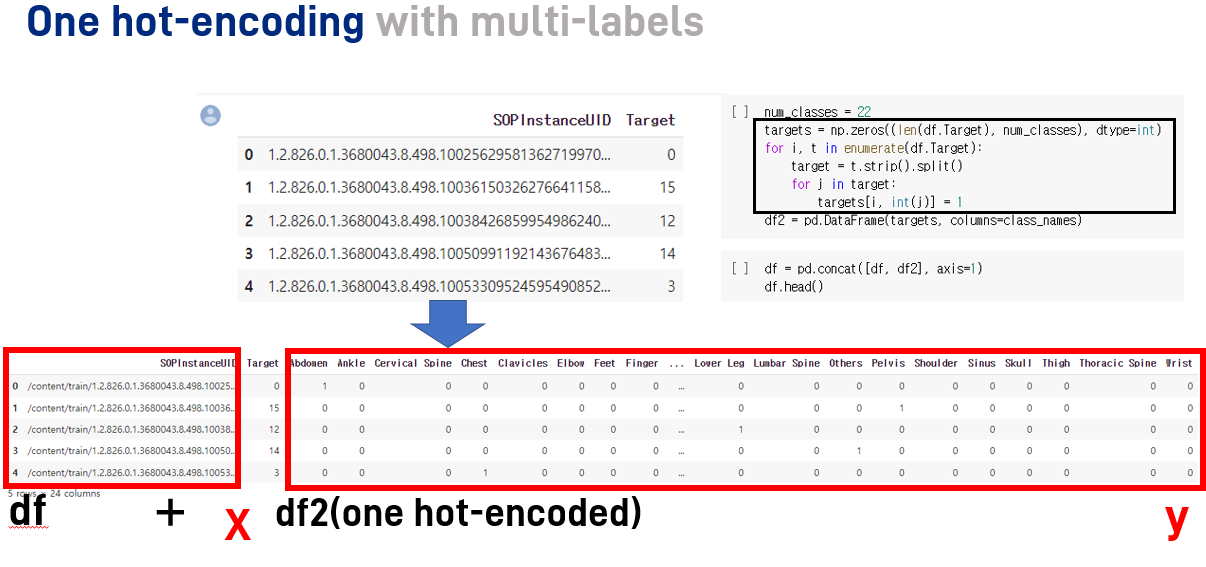
The one on the right is labeled as ‘hand’ and ‘wrist’. Similarly, the bottom left is labeled as three different body parts, ‘ankle’, ‘knee’, and ‘lower leg’. So, our competition is one of multilabel classification problems which is slightly different from what we learned in the Deep Learning lecture.

‘multiclass’ and ‘multilabel’



Multiclass classification is a classification that assigns to each sample one and only one label. Whereas the multi-label classification assigns to each sample a set of target labels. classifies each sample to one label among more than two classes. So, in multi-label classification, a sample can be assigned two or more labels.

**3. Data Preprocessing**



For data preprocessing, we transformed the ‘Target’ column into one hot encoded data, as this column contains multiple labels.

Explanation about one-hot encoding

Categorical data refers to variables that are made up of label values, for example, a “color” variable could have the values “red“, “blue, and “green”. Think of values like different categories that sometimes have a natural ordering to them.

Some machine learning algorithms can work directly with categorical data depending on implementation, such as a decision tree, but most require any inputs or outputs variables to be a number, or numeric in value. This means that any categorical data must be mapped to integers.

One hot encoding is one method of converting data to prepare it for an algorithm and get a better prediction. With one-hot, we convert each categorical value into a new categorical column and assign a binary value of 1 or 0 to those columns. Each integer value is represented as a binary vector. All the values are zero, and the index is marked with a 1.

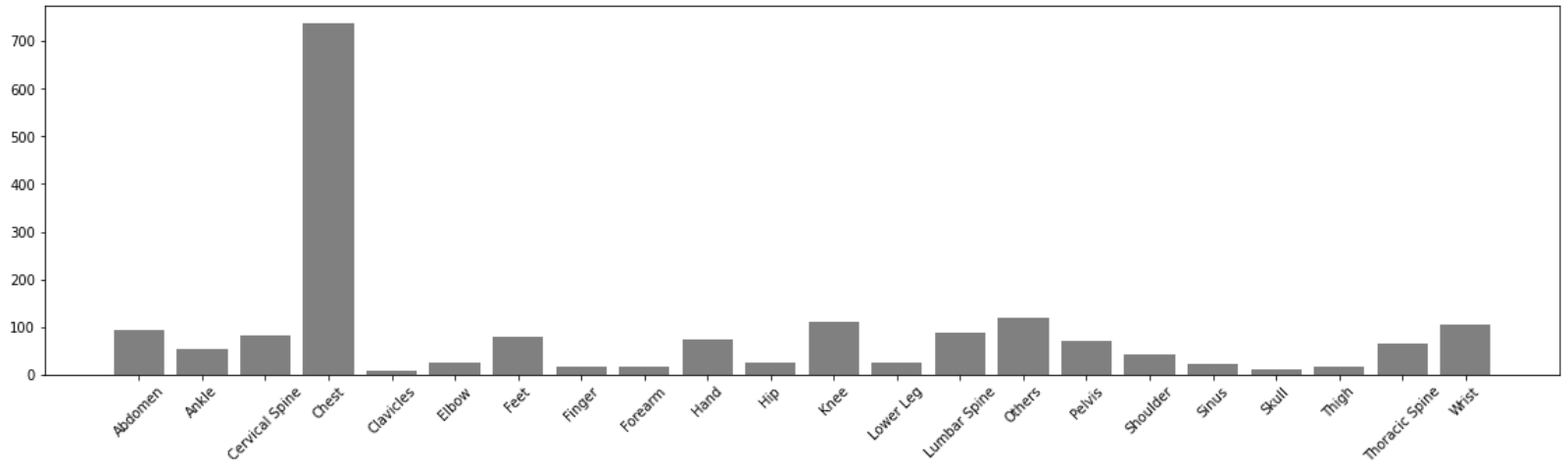
**4. Modeling**

Two pre-work conducted for better model, cross validation and data augmentation.

**4-0. Cross validation& Data augmentation**

We considered between stratified K Fold and K Fold which are methods of cross validation.

We focused on the distribution of data.



Distribution looks too skewed because data contains certain labels a lot. Even some data are less than ten.

So, we chose stratified K Fold which reflects original data’s distribution on each fold.

텍스트이(가) 표시된 사진

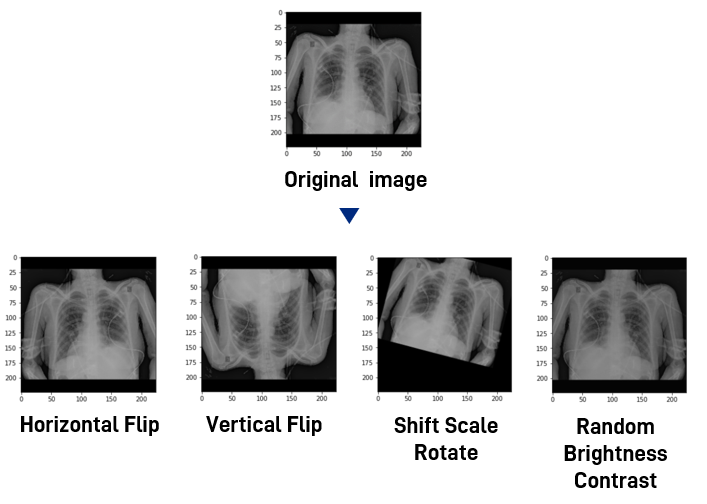
자동 생성된 설명

This is our code for stratified K Fold and train size is 1304 and valid size is 434.

**Data augmentation**

**텍스트이(가) 표시된 사진

자동 생성된 설명**



We conducted data augmentation, which increase the amount of data, to increase the effectiveness of model.

We worked in several types, for example, horizontal flip, vertical flip, shift scale rotate, random brightness contrast.

After two pre-work, we made two types of model by transfer learning.

**4-1. MODEL1: VGG 19**

테이블이(가) 표시된 사진

자동 생성된 설명

In CNN algorithm, there are several algorithms of image classification and VGGNET is one of them.

The type of model in VGGNET is determined by the number of layers. VGG19, we chose for our model, has 19 layers which are consisted of 16 convolutional layers and 3 fully connected layers. Type E in this table is VGG 19. VGG 19 has a 144 million parameters.

Architecture of VGG19

-A fixed size of (224 \* 224) RGB image was given as input to this network which means that the matrix was of shape (224,224,3).

-The only preprocessing that was done is that they subtracted the mean RGB value from each pixel, computed over the whole training set.

-Used kernels of (3 \* 3) size with a stride size of 1 pixel, this enabled them to cover the whole notion of the image.

-Spatial padding was used to preserve the spatial resolution of the image.

-Max pooling was performed over a 2 \* 2 pixel windows with stride 2.

-This was followed by Rectified linear unit(ReLu) to introduce non-linearity to make the model classify better and to improve computational time as the previous models used tanh or sigmoid functions this proved much better than those.

-Implemented three fully connected layers from which first two were of size 4096 and after that a layer with 1000 channels for 1000-way ILSVRC classification and the final layer is a softmax function.

Our code structure of VGG19

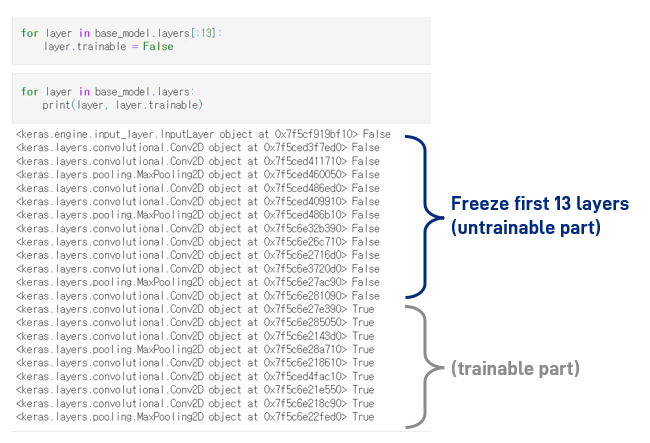
텍스트이(가) 표시된 사진

자동 생성된 설명

테이블이(가) 표시된 사진

자동 생성된 설명

To model VGG19, we divided layers of our pertained model into trainable group and untrainable group. We freeze first 13 layers for untrainable part and the rest are model fitted for trainable part.



After the code of pretrained model with VGG19, we add several Dense, dropout, and flatten layers.

텍스트이(가) 표시된 사진

자동 생성된 설명

테이블이(가) 표시된 사진

자동 생성된 설명

We set rectified adam for optimizer, binary cross entropy for loss, binary accuracy for metrics, and early stopping for callbacks.

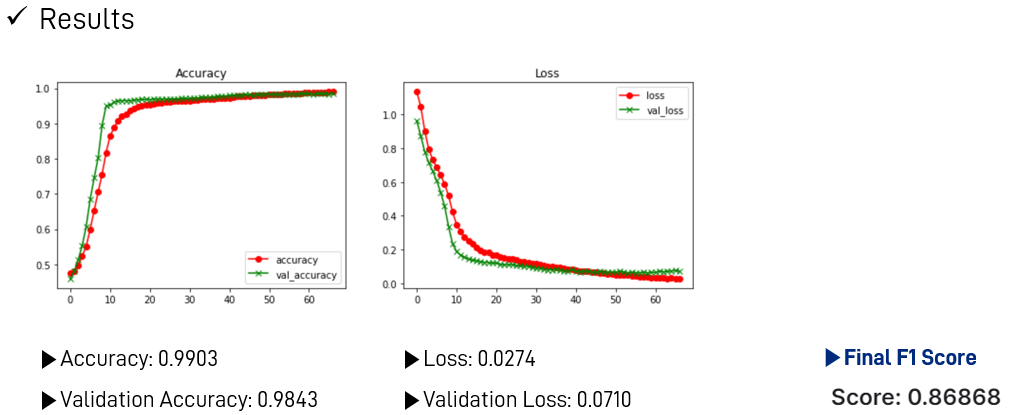
텍스트이(가) 표시된 사진

자동 생성된 설명

Explanation about Rectified Adam optimizer

Rectified Adam is a variant of the Adam stochastic optimizer that introduces a term to rectify the variance of the adaptive learning rate. It seeks to tackle the bad convergence problem suffered by Adam. The authors argue that the root cause of this behavior is that the adaptive learning rate has undesirably large variance in the early stage of model training, due to the limited amount of training samples being used. Thus, to reduce such variance, it is better to use smaller learning rates in the first few epochs of training - which justifies the warmup heuristic. This heuristic motivates RAdam which rectifies the variance problem:

Our results for VGG 19 model.



Accuracy is 0.9903 and validation accuracy is 0.9843.

Loss is 0.0274 and validation loss is 0.0710.

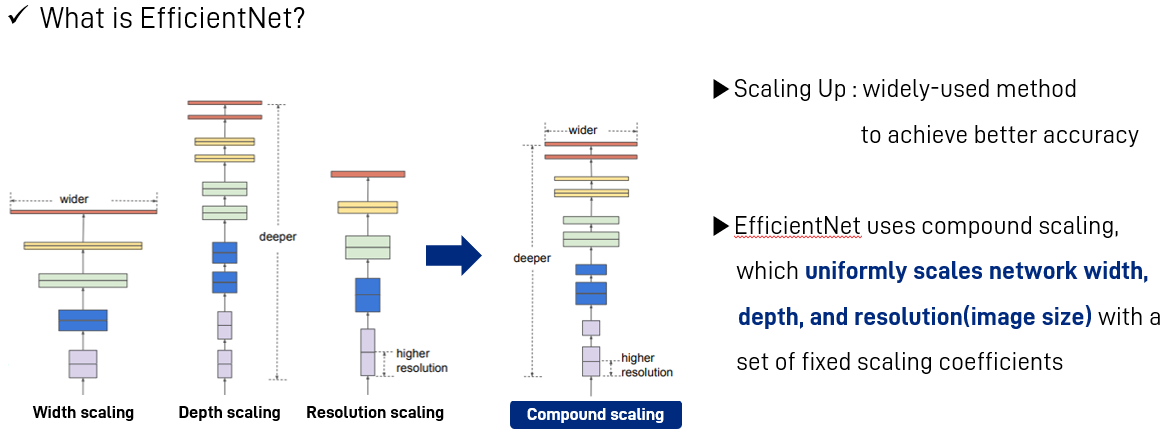
In this VGG19 model, we made 0.86868 for final F1 score.

**4-2. MODEL2: Efficient net B0**

The other model that we use as base model for transfer learning is Efficient net B0.

Efficient net was proposed in 2019 to achieve better accuracy and efficiency rather than other convolution nets which were used commonly before. Scaling up is basic principle to achieve better performance of model, and there are 3 big ways to scaling.

* Width scaling: to extend the width of channels; to increase the number of filters
* Depth scaling: to increase the number of layers
* Resolution Scaling: to enlarge the image size



But, efficient net uses **compound scaling**, which uniformly scales width, depth, and image size with a set of fixed scaling coefficients. The picture below shows the principle of calculating coefficients and how they work. Compound scaling shows higher accuracy than scaling only one factor.

텍스트이(가) 표시된 사진

자동 생성된 설명

Actually, efficient net has 8 types, from b0 to b7. The variable that determines the type is φ(phi). φ is a user-specified coefficient that controls how many more resources are available for model scaling. It means that when φ becomes greater (number in ‘b\_’ becomes larger), the model becomes complicated. We chose b0 model, which is the simplest model among them.

**테이블이(가) 표시된 사진

자동 생성된 설명**

The table above shows how efficient net B0 works compared with other convolution nets.

There are some metrics.

* Top-1 Accuracy: the probability that model prediction, which is the one with the highest

probability, must be exactly the expected answer

* Top-5 Accuracy: the probability that model’s top 5 highest probability answers match with

the expected answer

* Parameters: When models’ accuracies are similar, small number of parameters are better.
* FLOPs: FLoating point OPerationS, absolute amount of calculation. When models’ accuracies

are similar, small number of FLOPs are better.

We can see efficient net b0 has much smaller number of parameters and FLOPs.

But it shows almost same, or even better results than ResNet-50 and DenseNet-169.

**테이블이(가) 표시된 사진

자동 생성된 설명**

This table is brief structure of Efficient net b0 model.

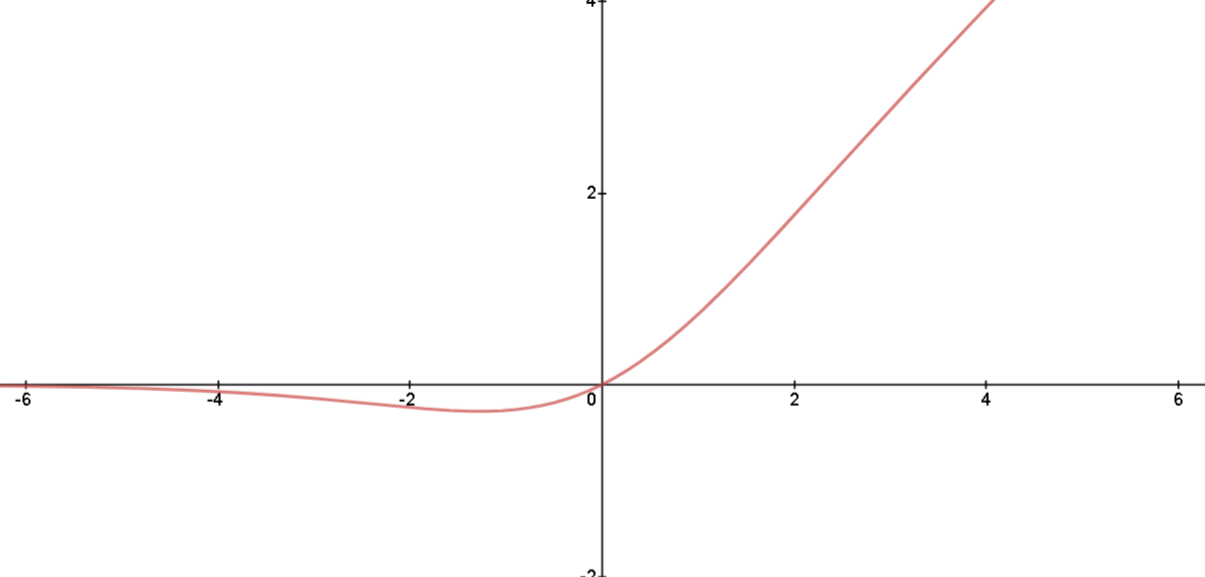
Actual model has much longer layers so it’s hard to show all of them.

B0 is mobile sized architecture having 11M trainable parameters. One can see that architecture uses 7 inverted residual blocks but each is having different settings. These blocks also use squeeze & excitation block along with swish activation.

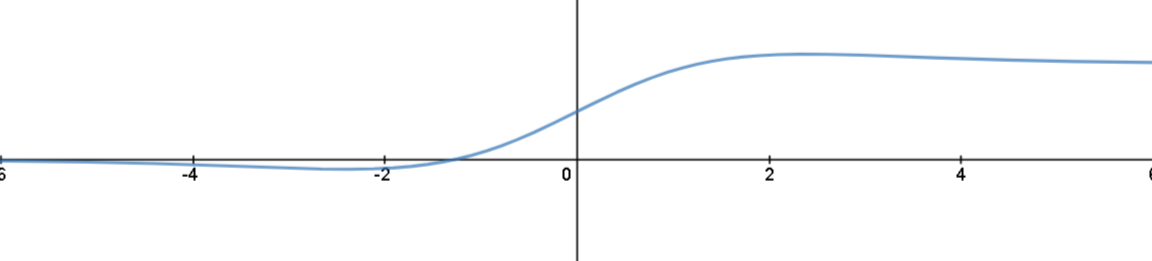
-Swish Activation

ReLu works pretty well, but it got a problem, it nullifies negative values and thus derivatives are zero for all negative values. There are many known alternatives to tackle this problem like leaky ReLu, Elu, Selu etc., but none of them has proven consistent. So Google Brain team suggested a newer activation that tends to work better for deeper networks than ReLU which is a Swish activation. They proved that if we replace Swish with ReLu on InceptionResNetV2, we can achieve 0.6% more accuracy on ImageNet dataset.

Swish(x) = x \* sigmoid(x)



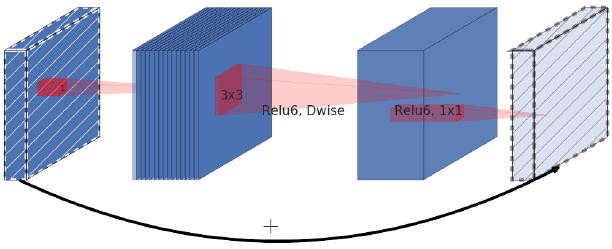
It’s gradient looks as shown in the below image



-Inverted Residual Block

The idea of a residual block was introduced in MobileNet architecture. MobileNet uses depthwise separable convolution inside the residual block which uses depthwise convolution first and then pointwise convolution. This approach decreases trainable parameters by a large number.

In an original residual block in ResNet, skip connections are used to connect wide layers and there are fewer numbers of channels inside a block. The inverted residual block does the opposite, skip connections connects narrow layers while wider layers are between skip connections.



-Squeeze and Excitation Block

When CNN creates output feature map from a convolutional layer, it gives equal weightage to each of channels. Squeeze and excitation block is a method to give weightage to each channel instead of treating them all equally.

SE block gives the output of shape (1 x 1 x channels) which specifies the weightage for each channel and the great thing is that neural network can learn this weightage by itself like other parameters.

Performance of EfficientNets family on ImageNet dataset.

And EfficientNet Uses 7 MBConvblocks. MBConv block takes two inputs, first is data and the other is block arguments. The data is output from the last layer. A block argument is a collection of attributes to be used inside an MBConv block like input filters, output filters, expansion ratio, squeeze ratio etc.

Our model’s brief shape of layers.

테이블이(가) 표시된 사진

자동 생성된 설명

…

테이블이(가) 표시된 사진

자동 생성된 설명

After the code of learning Efficient Net B0 as pretrained model, we add globalAveragePooling2D layer.

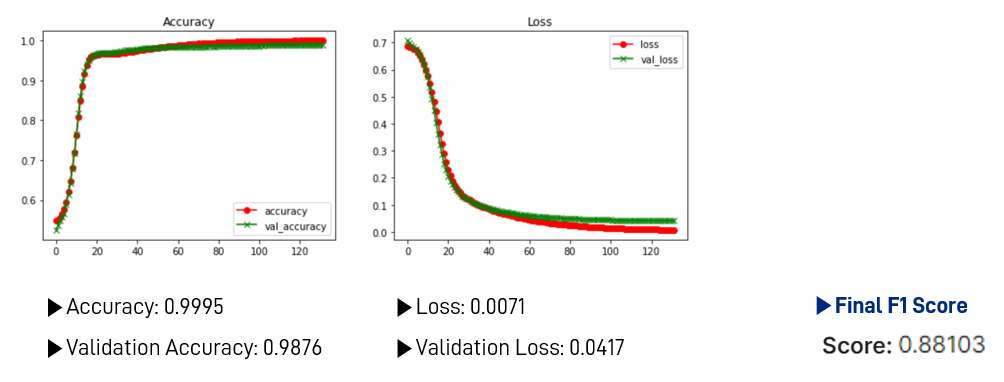
텍스트이(가) 표시된 사진

자동 생성된 설명

테이블이(가) 표시된 사진

자동 생성된 설명

And we use same optimizer, loss function, metric, and callbacks with the model including VGG19.



These are the results of our model fitting. Validation binary accuracy converges to about 99%.

And then we predict the label of test data, putting the predicted label on csv file, and submit it.

We get 0.88103 of F1 score, and this brings best score we’ve ever tried through various models.

**5. After participating this Competition**

We are very glad to finish our project. We worried a lot because almost members are beginners in deep learning, and regarded the competition as a giant barrier which we have to cross. Now, we think participation in this competition was meaningful because we can utilize what we learned from this deep learning class, not incomprehensive codes.

But there is something questionable about data that we use.

텍스트, 모니터, 실내, 텔레비전이(가) 표시된 사진

자동 생성된 설명

The pictures above do not look that different, but they were labeled differently. Left picture was labeled as lumbar and thoracic spine, and right picture was labeled as chest. We wonder if the algorithm learned minute differences between them. Actually, model looked confused between the set of hand and wrist, and the set of chest, lumbar spine, and thoracic spine.

We did further think about the reason. First, the train data is only 1738 data, and 40% of data is ‘chest’ data. We think small and skewed datasets disturb on learning how to distinct confusing pictures clearly.

Also, our lack of medical knowledge can be the other reason for this. DICOM image contains information about patients, and where the doctor should carefully look: the onset of disease. Normal people like us don’t notice the important point on the picture well. If we find it and adjust the model, it seems to get better performance. But this is not a medical class, and that content is beyond the deep learning class.

This point is the only thing that we pointed out as a problem on this competition.