A comparison of two convolutional neural networks for landmarks detection

LE Van Linh and BEURTON-AIMAR Marie

October, 2017

Abstract

In this study, we presented a comparison between two convolution neural networks (CNNs) which are used to predict the landmarks on biological images. The first model was presented by Celia Cintas et al[1]. The model is used to predict the 45 landmarks on the human ear. The second model is proposed by us as a result of tutorial about Lassagne framework[2]. The proposed model is designed to predict 8 landmarks on beetle's pronotum. Both models have experimented on the dataset of pronotum images and have implemented by Python on Lassagne framework. Besides, we also present another way to augment the data for CNN.

1 The models

In this section, we will describe the architecture of the models and the parameters that used during training and validation processes.

1.1 Model 1: Automatic ear landmarks detection

1.1.1 Architecture

Celia Cintas et al[1] proposed a method based on geometric morphometric and deep learning for automatic ear detection and feature extraction in the form of landmarks. The convolutional neural network was trained with a set of manually landmarks examples. The network is able to provide the morphometric landmarks on ear image automatically.

Three models were designed and trained for performing the automatic landmarks task. These architectures are different in the number of convolution layers, the filter sizes, and the learning rate. The data which used by the network for training and validation is a set of gray-scale images of the size 96×96 and set of list of manual landmarks coresponding to all the images in the image dataset.

Fig.1 shows the best architecture of three models. In this architecture, a structure of two convolutional layers with the filters, followed by maximum pooling and dropout layer. This structure is repeated **three times** to obtain features at different levels with different size of filters(i.e 4×4 and 3×3). After extraction the features, two fully connected linear layers with 1500 units each and a dropout layer is hired. The output layer contains 90 output units corresponding with 45 landmarks for the predicted position of the landmarks. But in our case, the output of the last layer has changed from 90 to 16 for adapting with the number of landmarks on pronotum.

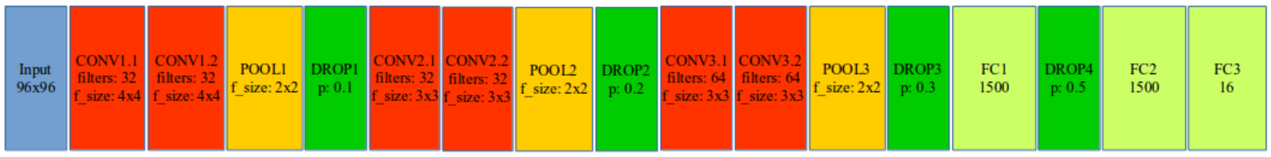


Figure 1: The best architecture for automatic ear's landmarks detection

1.1.2 Parameters

The network was trained with 5000 `epochs` and `batch size` of 128. For each epoch, the dataset is randomly split into the training set and validation set. The number of images in the sets is divided with the ratio of 80% : 20%, respectively. The `learning rate` is set to 0.03; and the initial of `momentum` is 0.9. During training, the learning rate and momentum are re-calculated to adjust with the number of remaining epochs. All the parameters in model 1 are shown in Table 1:

Parameter	Initial value	End value
Epochs	5000	
Training batch size	128	
Testing batch size	128	
Learning rate	0.03	0.00001
Momentum	0.9	0.9999

Table 1: The network parameters in the Celia model

1.2 Model 2: Automatic beetle landmarks detection

1.2.1 Architecture

From the tutorial of Daniel Nouri[1] about using CNN to detect facial key points. We propose a CNN to detect the landmarks on pronotum. The proposed network includes three convolutional layers followed by three maximum pooling layers and three full connected layers(Fig.2). The network receives the gray-scale image (256×192) as the input. The deep of convolutional layers is increased from $32, 64$, to 128 with different size of filter. The size of filters in pooling layers are kept in the same size of 2×2. At the end of network, three full-connected layers with the size of $500, 500$, and 16 are set up to predict the positions of landmarks. Besides, the model is designed with a small sharing learning-rate and the momentum. The learning-rate and the momentum are changed overtime of training.

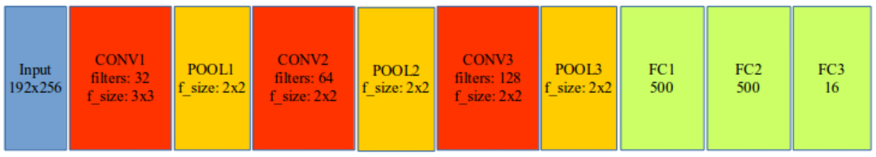


Figure 2: The architecture of proposed model

1.2.2 Parameters

The parameters in model 2 are shown in the Table 2:

[1]http://danielnouri.org/

Parameter	Initial value	End value
Epochs	5000	
Training batch size	128	
Testing batch size	128	
Learning rate	0.03	0.0001
Momentum	0.9	0.9999

Table 2: The network parameters in proposed model

2 Data

The experiment data includes 293 color images of pronotum. The images are divided into 3 subsets: training(200 images), validation(60 images) and testing set(33 images). In which, the training and validation are combined to use as the input data of the networks(total 260 images) during training. The images in testing set are used to evaluate the model. The images are chosen randomly to put into each set (after we have done some experiments).

Because the number of the images are limited (just 260 color images), it does not enough to use for training process. Additional, the models are worked on gray-scale images. So, we applied some rules to enlarge the dataset. The first rule is adding a constant value to a channel of RGB image, we will have a new RGB image. For example, from an original RGB image, if we add 10 to red channel, we will have a new image $(R+10)GB$. Then, we apply the same rule with blue and green channel, we will obtain two new images: $R(G+10)B$ and $RG(B+10)$. By that way, from an RGB image, we can generate three RGB images by adding a constant to each channel(each time just change to a channel). The second rule is splitting the channels of RGB image (because the models work on gray-scale). It means that we can generate six versions from an original image. At the end, the number of the image in the training data is $260 \times 7 = 1820$ images (six versions and original). Before giving to the models, the images are down-sampled with the size of 256×192. The number of the images in training set and validation set are splitted automatically by the model's parameter.

3 Experiments

In practical, convergence is usually faster if the average of each input variable over the training set is close to zero. Because the values of the pixels and the coordinates of the landmarks are positive. If we consider that we stay at the a layer of the network, and the weights are updated by an amount proportional to δx(δ is the scalar error at the layer and x is the input vector). When the input vectors are positive, the updates of weights that feed into the layer will be the same sign($sign(\delta)$), it means that the weights can only all decrease or all increase together for a given input. That, if the weight vector change direction, it can only do by zigzagging which is inefficient and thus slow down learning. Therefore, it is good to shift the inputs so that the average over the training set is close to zero. Moreover, when the input is set closed with zero, it will more suitable with the sigmoid activation function[3].

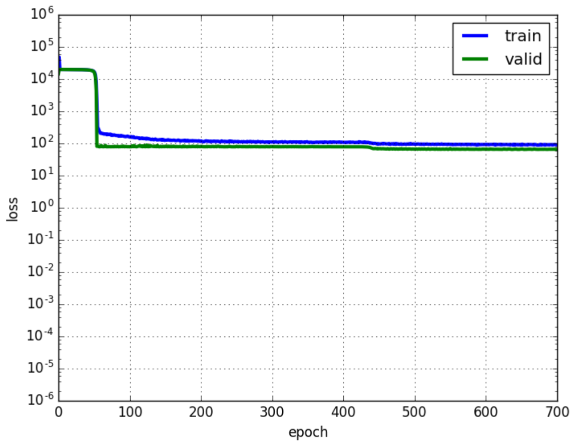
In this section, we describe the experimental processes of two models on pronotum dataset (section 2). The experiments were conducted in the way pre-processing data before giving to the network. At the first experiment, the values of the images are normalized and the coordinates of the landmarks are kept as original. In the second experiment, both inputs are normalized before giving to the network. Then, some improvements are added into model 2 to obtain the better result.

3.1 Experiment 1

In the first experiment, the images are kept in gray-scale with the size of 256×192. The brightness of the image is normalized to $[0, 1]$, instead of $[0, 255]$ while the coordinates of the landmarks are kept as original.

3.1.1 With model 1

We kept the same initial parameters of the model to train with the pronotum dataset. During training, the network cannot detect the loss of train and validation (*nan* value). A solution is given that we decreased the initial value (from 0.03 to 0.000001) and the stop value (from 0.0001 to 0.0000001) of learning rate. The network is re-trained with new parameters and we have succeeded. Fig.3a shows the first 700 epochs during training and validation process. We can see that the losses are stable and did not have much change after the 100^{th} epoch (to the end). Fig.3b shows the prediction landmarks on an image in test set (the network never saw before). The predicted landmarks are closed with the pronotum but their location is still inaccurate.



(a) Training losses and validation loss

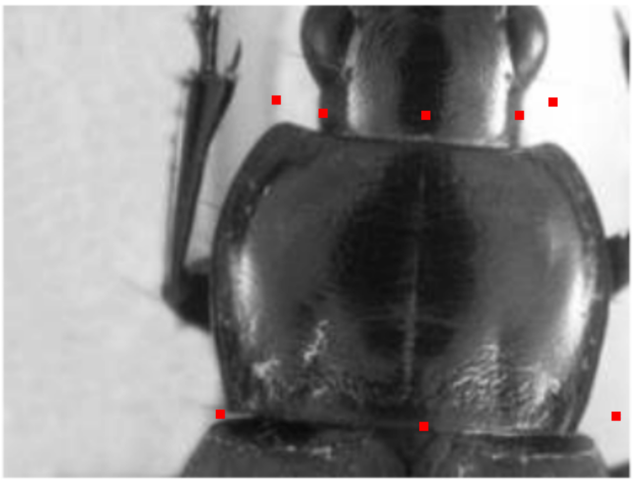
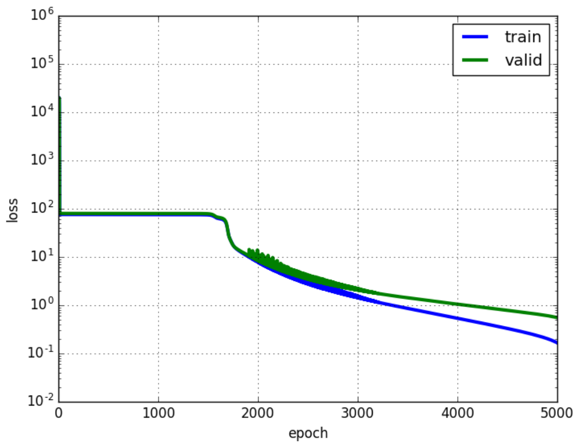

(b) A pronotum with predicted landmarks

Figure 3: The evaluation of pronotum data on model 1

3.1.2 With model 2

From the experiment of model 1 on pronotum data. We change the learning rate in the model 2 before training the network: initial value is changed from 0.3 to 0.00001 and stop value is changed from 0.0001 to 0.000001. The training loss and validation loss are shown in Fig.4a. The loss is stable to 1600 epochs but then, they decreased to the end: 0.16694 for training loss and 0.55584 for validation loss. Clearly, when we compare the losses from two models, a difference has appeared, and the results of model 2 are worth considering. Fig.4b shown the predicted landmarks on a test image. The model 2 is used to predict the landmarks on the test set (includes 33 images). Then, the correlation coefficient between manual and predicted landmarks is computed by using different correlation methods[4, 5, 6]. The correlation results are shown in Table.3



(a) Training losses and validation loss

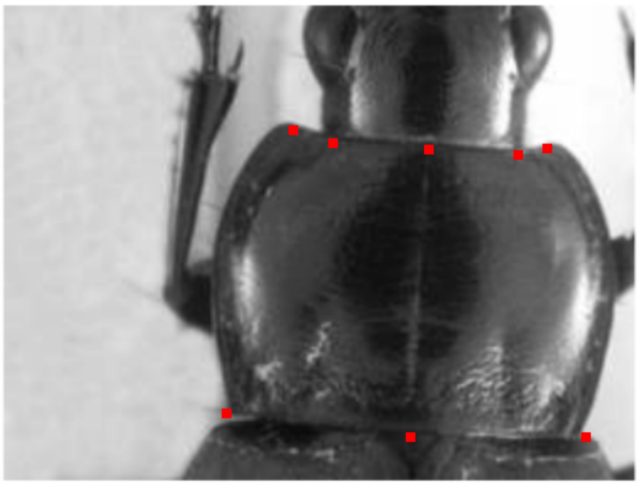

(b) A pronotum with predicted landmarks

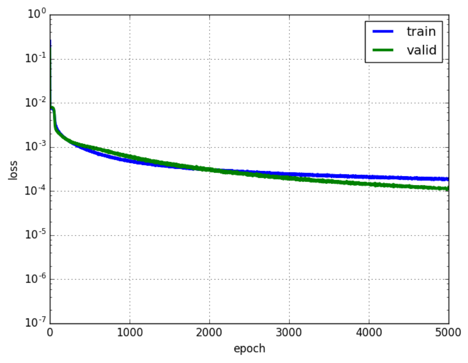
Figure 4: The evaluation of pronotum data on model 2

Method	x correlation	y correlation
Pearson	0.9966784	0.9957729
Spearman	0.9913126	0.9565425
Kendall	0.9273012	0.8273057

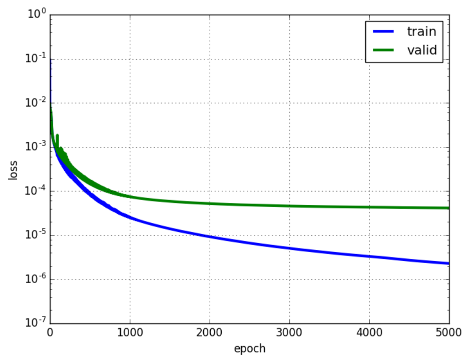
Table 3: The correlation between manual and predicted landmarks on test images

3.2 Experiment 2

In the second experiment, both the image values and the target (landmark coordinates) are normalized to a range of $[-1, 1]$, instead of $[0, 256]$ and $[0, 192]$ before giving to the networks[3]. After some trials, the learning rates are set at the beginning to increase the speed of the processes. Fig.5 show the loss during training and validation of two models with the new version of data. We can see that the losses from model 2 is really better than from model 1. But, at the end of training, the overfitting has appeared in model 2.



(a) Model 1



(b) Model 2

Figure 5: The training and validation loss on two models

To prevent the overfitting on the model 2, we have changed the ratio of train/val selection. The new ratio is set to 60% : 40%, instead of 80% : 20% as the beginning. The network is re-train with the new data but the overfitting is also not stopped. Then, we decide to modify the structure of the network (Fig.6):

- Increasing the number of output at the last two hidden layers (full-connected) from 500 to 1000, but the result did not lead to better performances,
- Then, we added four dropout layers to the network. The dropout layer is considered as the good solution to prevent the overfitting[7]. They have been located following the pooling layers and the first fill-connected layer. The dropout ratios are 0.1, 0.2, 0.3 and 0.5.

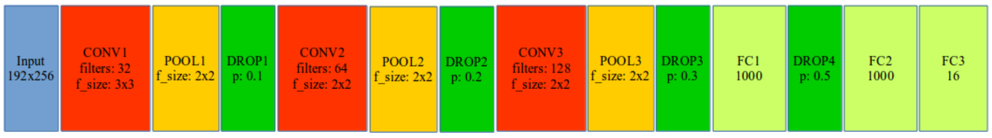


Figure 6: The proposed model with dropout layers

Fig.7 shows the losses after the model have been modified. The overfitting problem has been solved but the losses are stability from 2000^{th} until the end(we need more data).

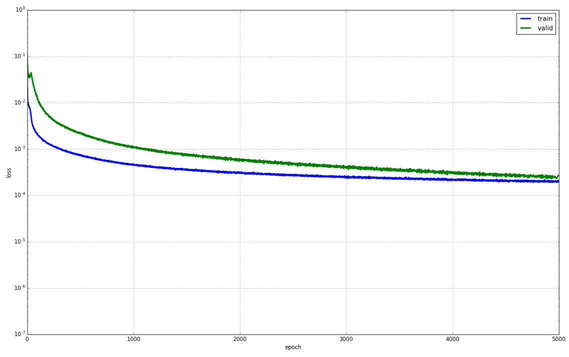
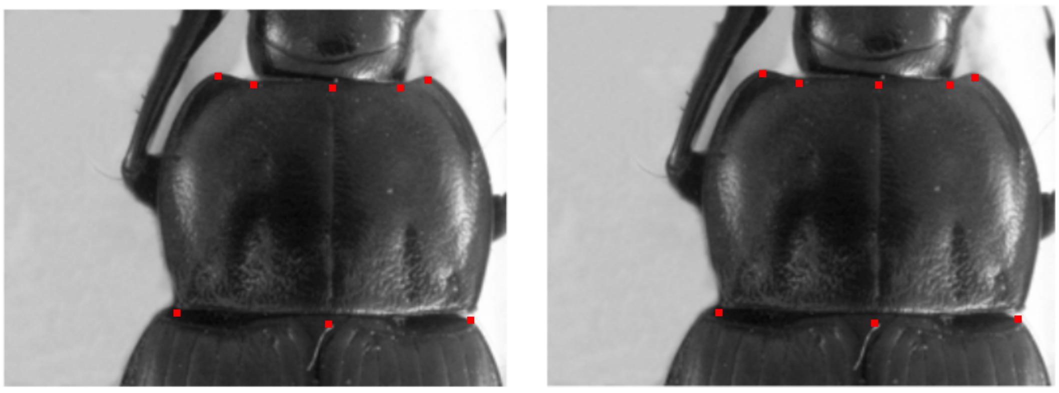


Figure 7: The training and validation loss with dropout layers

Fig.8 and show the predicted landmarks on an image in test dataset.



(a) Model 1 (b) Model 2

Figure 8: The predicted landmarks on a test image

Both models are evaluated on test set. Table 4 shows the correlation coefficient of model 1 and Table 5 shows the correlation coefficient of model 2. Clearly that, the result from model 2 have been improved a little bit when we compare with the last result(Table 3). While we do not see the difference between the results of model 1 and model 2. The only difference is the architecture of the models: model 1 is deeper than model 2 that means the model 1 takes more time to train than the model 2.

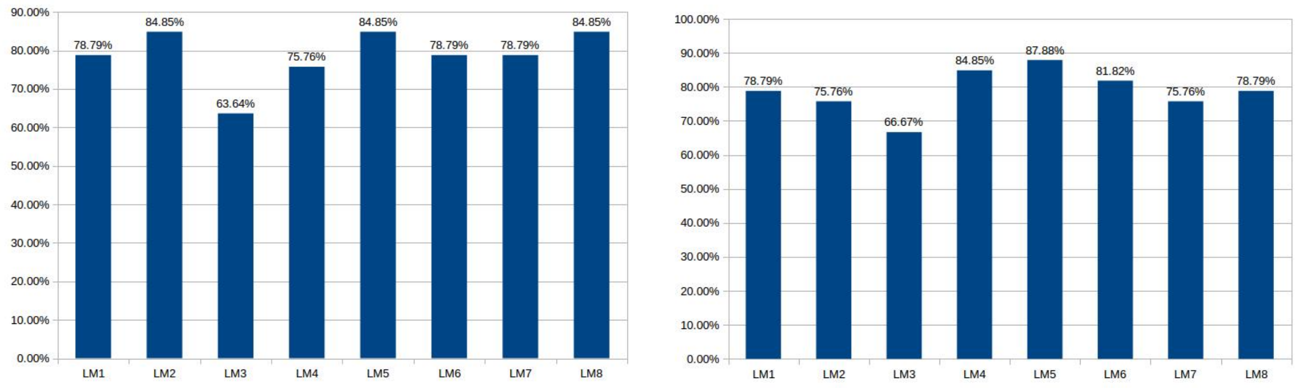
Method	x correlation	y correlation
Pearson	0.9976008	0.9983418
Spearman	0.9948408	0.9875125
Kendall	0.9463506	0.9133915

Table 4: The correlation between manual and predicted landmarks on pronotum images with model 1

Method	x correlation	y correlation
Pearson	0.9970585	0.9978605
Spearman	0.9942475	0.9859642
Kendall	0.9430501	0.9067739

Table 5: The correlation between manual and predicted landmarks on pronotum images with model 2

Besides the evaluation on correlation coefficient, we also consider the accuracy on each landmark of all images in the test set. The distance between manual landmark and predicted landmark is calculated. Then, the standard deviation(SD) is used to quantify the dispersion of a set of distances. Fig.9 show the statistic on each landmark of test dataset(from two models). All landmarks have been detected with an accuracy greater than 70% (excepts 3^{rd} landmark). The first model provided the good accuracy for $2^{nd}, 5^{th}, 8^{th}$ landmarks, while the second model had the good position for $4^{th}, 5^{th}, 6^{th}$ landmarks. Generally, we can see a vast difference between the correlation coefficient result and the proportion on each landmark.



(a) Model 1 (b) Model 2

Figure 9: The proportions of well predicted landmarks on each model

4 Conclusions

Two models that used to predict the landmarks on 2D gray-scale images are studied. For each case, the model is suitable with different dataset but the results are still good when we change the training data (the case of model 1 when we re-train on pronotum). Besides, we proposed a different way to enlarge the data in the case the dataset is limited. From the correlation coefficients of each model shows that the effects of two models are similar. Additional, if we consider on the statistic side, the coefficients are enough good to precise. But, when we see the predicted landmarks on the images, we also have some case that is not good as we expect.

References

[1] Celia Cintas, Mirsha Quinto-Sánchez, Victor Acuña, Carolina Paschetta, Soledad de Azevedo, Caio Cesar Silva de Cerqueira, Virginia Ramallo, Carla Gallo, Giovanni Poletti, Maria Catira Bortolini, et al. Automatic ear detection and feature extraction using geometric morphometrics and convolutional neural networks. *IET Biometrics*, 6(3):211–223, 2016.

[2] Sander Dieleman, Jan Schlter, Colin Raffel, Eben Olson, Sren Kaae Snderby, Daniel Nouri, et al. Lasagne: First release., August 2015.

[3] Yann A LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–48. Springer, 2012.

[4] Julie Pallant. *SPSS survival manual.* McGraw-Hill Education (UK), 2013.

[5] Jerome L Myers, Arnold Well, and Robert Frederick Lorch. *Research design and statistical analysis.* Routledge, 2010.

[6] Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938.

[7] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 15(1):1929–1958, 2014.

A comparison on parameters of the networks

layers	model 1	model 2	model 2_1	model 2_end
input	$1 \times 96 \times 96$	$1 \times 256 \times 192$	$1 \times 256 \times 192$	$1 \times 256 \times 192$
layer 1	CONV(32,4,1,0)	CONV(32,3,1,0)	CONV(32,3,1,0)	CONV(32,3,1,0)
layer 2	CONV(32,4,1,0)	POOL(2,2,0)	POOL(2,2,0)	POOL(2,2,0)
layer 3	POOL(2,2,0)	CONV(64,2,1,0)	CONV(64,2,1,0)	**DROP(0.1)**
layer 4	DROP(0.1)	POOL(2,2,0)	POOL(2,2,0)	CONV(64,2,1,0)
layer 5	CONV(32,3,1,0)	CONV(128,2,1,0)	CONV(128,2,1,0)	POOL(2,2,0)
layer 6	CONV(32,3,1,0)	POOL(2,2,0)	POOL(2,2,0)	**DROP(0.2)**
layer 7	POOL(2,2,0)	FC(500)	FC(**1000**)	CONV(128,2,1,0)
layer 8	DROP(0.2)	FC(500)	FC(**1000**)	POOL(2,2,0)
layer 9	CONV(64,3,1,0)	FC(16)	FC(16)	**DROP(0.3)**
layer 10	CONV(64,3,1,0)	-	-	FC(1000)
layer 11	POOL(2,2,0)	-	-	**DROP(0.5)**
layer 12	DROP(0.3)	-	-	FC(1000)
layer 13	FC(1500)	-	-	FC(16)
layer 14	DROP(0.5)	-	-	-
layer 15	FC(1500)	-	-	-
layer 16	FC(16)	-	-	-

Table 6: The parameters at each layer of the models

Which:

- CONV(x,y,z,t): convolutional layer with the parameters: *x = number of filters, y = size of filter matrix, z = stride value, t = padding value*
- POOL(y,z,t): maximum pooling layer with: *y = size of filter, z = stride value, t = padding value*
- DROP(p): dropout layer with *p is the dropout ratio*
- FC(x): full-connected layer with *x is the number of output*