Landmarks detection by applying Deep networks

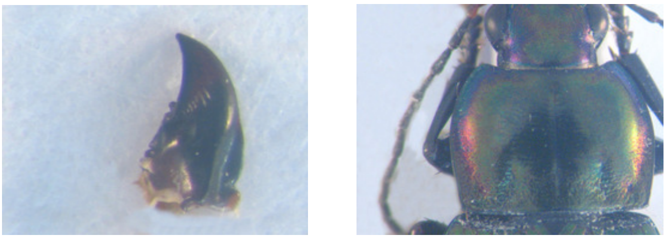
Abstract—**Morphometric analysis is a general method applied to organisms and used to appreciate the covariances between the ecological factors and the organisms (shape, size, form,...) in which, landmark-based morphometry is known as one of the approaches to analyze the characteristics of organisms. Finding enough landmarks can give to biologists a comprehensive description of organism. In this study, we propose a convolutional neural network (CNN) to predict the landmarks on beetle images. The network is designed as a pipeline of layers, it was trained with a set of manually labelled landmarks dataset. Then, the network is used to provide the morphometric landmarks on biological images automatically. The coordinates of predicted landmarks are evaluated by computing the distance with the manual coordinates given by the biologists. Besides, the average of distance errors on each landmark has been also given. The network is implemented by Python on Lasagne framework.**

Index Terms—**Morphometry, biological, landmarks, deep learning, convolutional neural networks**

I. Introduction

Morphometry analysis refers to measure the topography of an object, a notion that includes the shape and size. Morphometry analysis is often applied to biological organisms. Biologists work with several parameters from organisms such as lengths, widths, masses, angles,... to analyze the interaction of environment with organisms development. Besides the traditional information, landmarks (or points of interest in the image) are known as one of the characteristics to analyze the shape. Instead of collecting all information, the shape is determined by a finite set of points, called landmarks. The landmarks are the points that store the important information about the shape of the object, *for example*, the corners of the human mouth are a kind of landmarks. Currently, the landmarks are along the outline of the object but in some special cases, it has been defined inside the object. Morphometry landmarks are a kind of points-of-interest, they are directly linked to the animal anatomy. In our study, the morphometric landmarks are specific points defined by the biologists. They are used in many biological studies. Manual landmarks setting is time-consuming and difficult to reproduce. Therefore, a method that gives the automatic location of landmarks could have a lot of interest.

This work introduces a method for automatic detection of the landmarks on biological images. The main idea consists design and train of a convolutional neural network [1] with a set of manual landmarks. By this way, the trained network will be able to detect the morphometry landmarks on biological images. The dataset includes 293 beetles images from Brittany lands. All the images are presented in RGB color with two dimensions. For each beetle, the biologists took images of five parts: *left and right mandibles, head, body, and pronotum*. For each image, a set of landmarks has been manually determined by experts. In another work [2], a method has been presented to determine the landmarks on left and right mandibles (Fig. 1a). This method is based on the image processing techniques [3], combining with principal component analysis [4] and SIFT descriptor [5]. In the context of this work, we work on the dataset of pronotum images (Fig. 1b). For each pronotum image, a set of 8 manual landmarks have been set by the biologists. The coordinates of manual landmarks were used as the input to train the network. During the first phase, a number of 260 images and their manual landmarks were used to train and validate the network. The remaining images were used to evaluate the output model of the network in the second phase.



(a) Right mandible

(b) Pronotum

Fig. 1: An example of right mandible and pronotum image

In the next section, we present related works to determine the landmarks on 2D images by CNN. In section 3, we present an overview about convolutional neural networks. Section 4 presents the architecture of the network, its parameters, and the implementation. All the experiments and analysing the results are detailed in section 5.

II. Related works

Landmark or point of interest is a specific point that may contain the useful information. For example, the tip of the nose or the corners of the mouth are landmarks on human face. In image processing, we can consider two kinds of cases: the object of interest can or not be segmented. Setting landmarks can not be achieved in the same way depending on which situation we are. When segmentation can be applied, Lowe et al. [5] have proposed SIFT method to find the corresponding keypoints between two images. Palaniswamy et al. [6] have proposed a method based on probabilistic Hough Transform to automatically locate the landmarks in digital images of Drosophila wings. Krahenbuhl et al. [2] have proposed a method which have been extended from Palaniswamy's method, to determine landmarks on mandibles of beetles. The mandibles of beetle have the simple shape and easy to segment. We have obtained good enough results about determining the landmarks automatically on mandibles. Unfortunately, this method can not be applied to other parts of beetles than the pronotum seems is segmentation has too many noises.

In recent years, deep learning is known as a solution in computer vision, especially image analysis. Using convolutional network to determine the landmarks on 2D images has achieved better results and it seems that good solutions for the images that can not segment. Yi Sun et al. [7] have proposed cascaded convolutional neural networks to predict the facial points of interest on the human face. Zhanpeng Zhang et al. [8] proposed a *Tasks-Constrained Deep Convolutional Network* to optimize facial landmarks detection. The model determines the facial landmarks with a set of related tasks such as head pose estimation, gender classification, age estimation, face recognition, or facial attribute inference. In biology field, Cintas et al. [9] has introduced a network to predict the landmarks on human ears. After training, the network has the ability to predict 45 landmarks on human ears. In this way, we have applied CNN computing to work with pronotum landmarks.

III. Convolutional neural networks

Deep learning allows computational model composed of multiple processing layers to learn representations of data with multiple levels of abstraction [10]. Each layer extracts the representation of the input data from the previous layer and computes a new representation for the next layer. In the hierarchy of model, higher layers of representation enlarge aspects of the input that is important for discrimination and suppress irrelevant variations. Each level of representations is corresponding to the different level of abstraction. During training, it uses gradient descent optimization method to update the learnable parameters via backpropagation. The development of deep learning opens promise results of the artificial intelligent on high dimensional data, therefore applicable to many domains: image recognition and classification [11]–[13], speech recognition [14]–[16], question answering [17] and language translation [18] [19].

Convolutional neural networks (CNNs) contribute to solve the state of the art in many computer vision tasks such as classification [11][12] or recognition [20][21]. A CNN consists a number of connected layers. The layers of CNN has neurons arranged in three dimensions: *width, height, and depth* with learnable parameters. Fig. 2 shows a common example of CNN. It is a pipeline of usual layers: convolutional layers (CONV), pooling layers (POOLING), dropout layers (DROPOUT), and full-connected layers (FC).

Convolutional layer computes a dot product between their weights and a small region in the input. At the output, the results of connected local regions are combined. Convolution layer uses a set of learnable filters as parameters. Each filter is small spatially but extends the depth of the input. *Pooling* layer is used to down-sampling the input, reduce the computational cost in remaining layers, and control overfit. *Dropout* layer refers to dropping out units in the network. By dropping a unit out, we mean temporarily removing it from the network, along with all its incoming and outgoing connections. *Full connected* layer refers to the output of the network. The number of outputs of the last full-connected layer are corresponding with the number of predicted values.

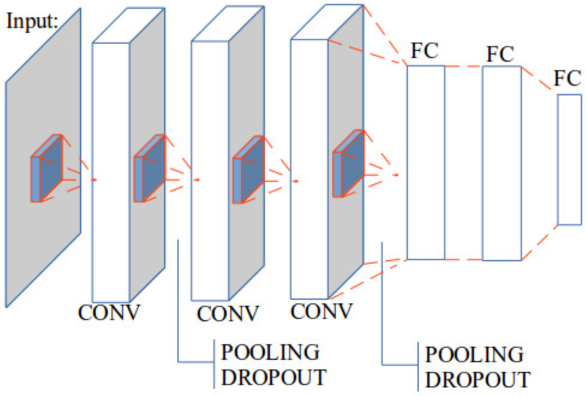


Fig. 2: An example of usual convolutional neural network

From the beginning of deep learning until now, many deep learning frameworks have been developed. Almost the framework was open source. According to the written programming languages, the frameworks can be separated into two main groups: C++, *such as Caffe, Deeplearning4j, Microsoft Cognitive Toolkit* and Python *i.e Keras, Theano, PyTorch*. However, along with the development of deep learning, the frameworks support to design the network with other languages.

Theano is an open source framework developed by machine learning group at the University of $Montréal$. It is a Python library that allows to define, optimize and evaluate mathematical expressions relating multi-dimensional arrays efficiently by using a Numpy package. Theano supports compiled on either CPU or GPU architectures. Lasagne [22] is a lightweight library in Theano. It allows to build and train the neural networks. In this work, we have used Lasagne to implement the proposed neural network. Recently, Theano has been stopped to develop but its community is still large. The networks which have been designed by Theano are still good to use in deep learning.

IV. Application to landmarks identification

In previous sections, we have had an overview of the landmarks and CNN. In this section, we describe the designing processes of the model. We have also presented the data preprocessing, training and evaluating the model.

A. Preprocessing data

The images come from a collection of 293 beetles from Brittany lands. All the images are taken with the same camera under same conditions with a 3264×2448 resolutions. For each specific part, a set of manual landmarks has been determined by biologists, *for examples, 8 landmarks for pronotum, 16 landmarks for the left mandible, 18 landmarks for right mandible*. In the content of this study, we work on pronotum part of beetle. The provided dataset contains 293 images, each image with 8 landmarks provided by biologists. The dataset was split into a training set with 260 images (training and validation) and a testing set of 33 images. During the training, the network learned the information through a pair of *(image, landmarks)* in training set. At the testing phase, the image without landmarks was given to the trained network and the predicted landmarks will be given at the output. Fig. 3 shows an example of pronotum image with its manual landmarks.

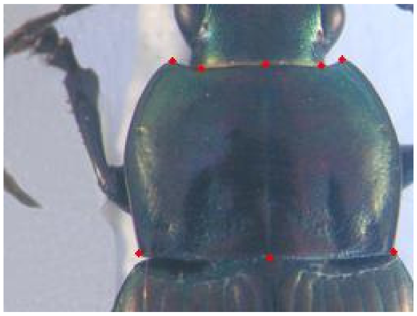


Fig. 3: An example of pronotum with manual landmarks

In some succeed networks [11][7][9], the maximum size of the inputs is not over 256 pixels. In our case, the resolution of the image is large, it becomes a difficulty for the network. During training and testing, the images are down-sampling to the new resolution of 256×192. Certainly, the landmark coordinates of the image are also scaled to suit their new resolution.

The proposed network has a large number of learnable parameters. In addition, the size of the dataset is limited, this means that overfitting will occur during the training process. Therefore, we need to enlarge the size of the dataset. In image processing, we usually apply transform procedure (i.e rotate, translate) to generate a new image but in fact, when we compute the value of the pixels, it does not change while CNN computes the values of the pixels. Therefore, we have applied two other procedures to increase the number of images in the dataset. To address this problem, we have applied two procedures to enlarge the size of the dataset.

The first procedure was applied to change the value of each channel in the original image. According to this, a constant is added to a channel of RGB image and for each time, we just change the value of one of three channels. For example, from an original RGB image, if we add a constant $c = 10$ to the red channel, we will obtain a new image with the values at red channel by greater than the red channel of original image a value of 10. By this way, we can generate three new RGB images from a RGB image.

The second procedure is splitting the channels of RGB images. It means that we separate the channels of RGB into three gray-scale images. This work seems promising because the network works on single-channel images. At the end, we can generate six versions from an image, the total number of images used to train and validate is $260 \times 7 = 1820$ images (six versions and original image). The number of images that used for training and validation is splitted randomly by a ratio (training: 80%, validation: 20%) that has been set during the network setup.

In practical, when we work with CNN, convergence is usually faster if the average of each input variable over the training set is close to zero. Moreover, when the input is set closed with zero, it will be more suitable with the sigmoid activation function [23]. According to [23], the brightness of the image is normalized to $[0, 1]$, instead of $[0, 255]$ and the coordinates of the landmarks are normalized to $[-1, 1]$, instead of $[0, 256]$ and $[0, 192]$ before giving to the network.

B. Network architecture and training

Three networks have been proposed and trained to perform the best architecture for automatically landmarking predictions. The networks consist on several common layers in the neural network with different learnable parameters They receive the same input of $1 \times 256 \times 192$ to train but they have different number of layers. In this section, we introduce the architectures of the networks and the process to improve the architecture from the beginning of designing.

Fig. 4 shows the architecture of the first network. This model has been designed from a generic method: the layers in the network are very common as the other neural networks. The network consists on three repeated-structure of a convolutional layer followed by a maximum pooling layer. The depth of convolutional layers increases from $32, 64$, and 128 with different size of the filter kernel: 3×3, 2×2, and 2×2. All the kernels of pooling layers have the same size of 2×2. At the end, three full connected layers have been added to the network. The outputs of the full connected layers are $500, 500$, and 16, respectively. The output of the last full-connected layer corresponds to 8 landmarks (x and y coordinates) which we would like to predict. The training result shows that the architecture of this model is not good enough to solve the problem. It is still simple and overfitting has appeared during training and validation.

The second network has the same architecture as the first one. But, number of outputs at full-connected layers have been modified from 500 to 1000 (the second model) to prevent the overfitting, but the result did not lead to better performances.

In the third network, instead of changing the parameters, we have added four dropout layers to the network. These are considered as the good solution to prevent the overfitting. The idea of dropout is to randomly drop units from the neural network during training. It prevents units from co-adapting too much. During training, dropout samples are done from an exponential number of different “thinned” network. At test times, it is easy to approximate the effect of averaging the prediction of all thinned networks by simply using a single unthinned network with smaller weights. This significantly reduces overfitting and gives major improvements over other regularization methods [24]. Fig.5 shows the architecture of the third network. The first three dropout layers are supplemented to the repeated-structures followed the maximum pooling layers. In that way, structure becomes a convolution

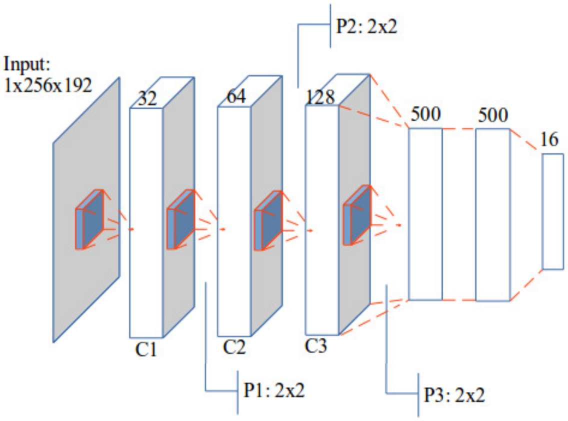


Fig. 4: The architecture of the first network

layer with square filter, followed by a *maximum* pooling and dropout layer. The probability values used for dropout layers are 0.1, 0.2, and 0.3. Actually, we keep the same value for the parameters of the convolutional $(32, 64$, and $128)$, pooling $(3\times3$, 2×2, and $2\times2)$ and full-connected layers $(1000, 1000$, and $16)$ as the second one.

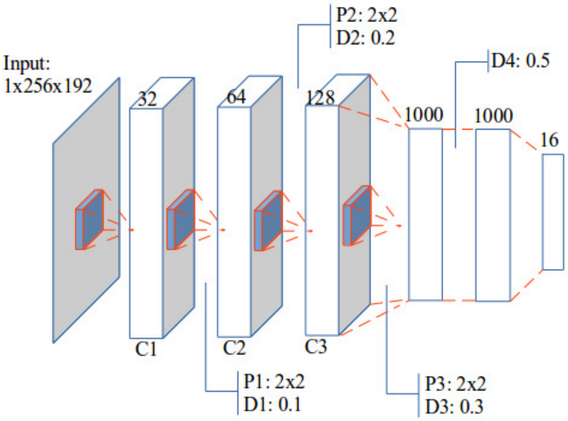


Fig. 5: The architecture of the last network

The remaining dropout layer is inserted between the first two full connected layers. The probability value of this layer is set to 0.5. The output layer still contains 16 units corresponding to the coordinates of 8 predicted landmarks.

Additionally, the network is designed with a small sharing learning rate and the momentum. As usual, they have been used to perform gradient descent during backward phase to update the parameters of the layers. The value of learning rate and momentum are updated over training time to fit with the remaining number of iterations. The implementation of this architecture used Python on Lasagne framework [22] which allows to train the network on GPU. The training process took around 3 hours using NVIDIA TITAN X cards. The design of the network is available on GitHub[1].

C. Results

The dataset has been built by the biologists. It includes the images and manual landmarks. So, we can use the manual landmarks coordinates as ground truth to evaluate the coordinates of predicted landmarks. In the context of deep learning, landmark prediction can be seen as a regression problem. Therefore, the quality metric is used to evaluate the results. In particular, we use root mean square error (RMSE) to compute the accuracy of the implemented architecture.

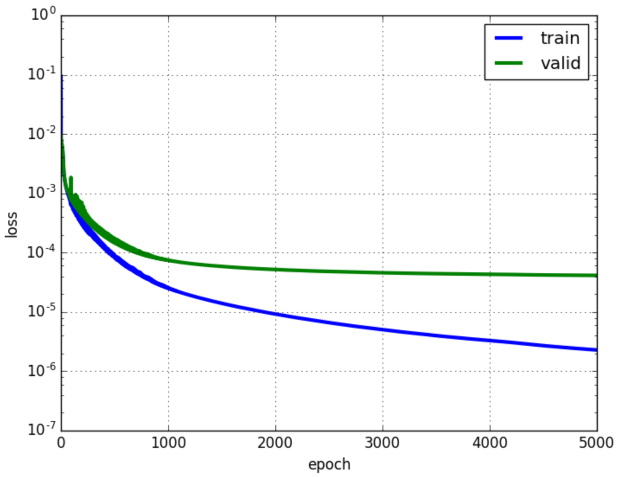


Fig. 6: Learning curves of the first model.

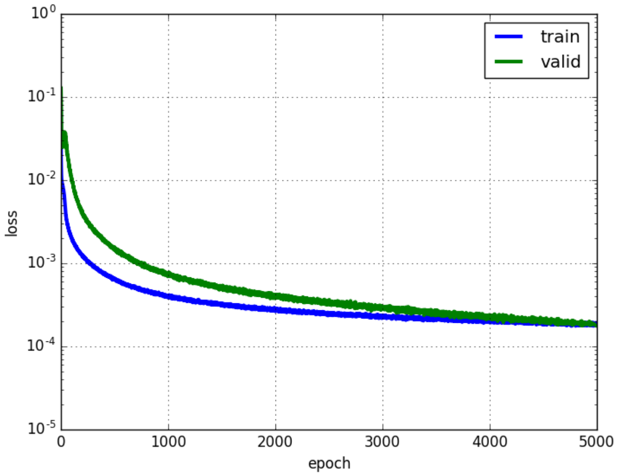


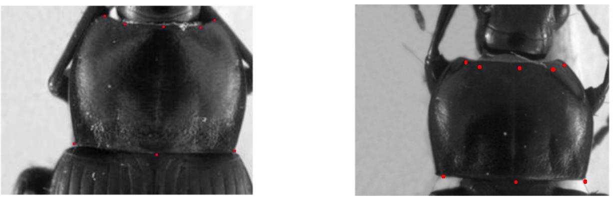
Fig. 7: Learning curves of the last model.

Fig.6 and 7 show the training errors and the validation errors of a training time on the first and the third model, respectively. The blue curves present RMSE on training set, the green curves present the validation errors. Clearly, the overfitting has appeared in the first model. In Fig.6, we can see that if

[1] It is freely obtained by request the authors.

the training is able to decrease with the number of epochs[2], it is not the case of validation loss. At the opposite in the third model, we can see some different values for the two losses at the beginning but after several epochs, these values become more proximate and the overfitting problem has been solved.

Fig.8 shows the predicted landmarks on test images set by the thrid model. When we consider the distance between the predicted and manual landmarks, the accuracy on coordinates of predicted landmarks on Fig.8a is 99%. The propotion on Fig.8b is 80%. Besides the losses of training, the distance from predicted landmarks to manual landmarks of the test images deserve attention also. Firstly, the distance between them is calculated. Then, the standard deviation [25] is used to quantify the dispersion of a set of distances. Table.I shows the average error distance given on each landmark.



(a) Image with well-predicted landmarks

(b) Image with inaccuracy landmarks

Fig. 8: The predicted landmarks on an image in test set (red points)

TABLE I: The average distance per landmark

#Landmark	**Distance** (in pixels)
1	4.002
2	4.4831
3	4.2959
4	4.3865
5	4.2925
6	5.3631
7	4.636
8	4.9363

Fig.9 shows the distribution of the distances on the first landmarks of all images. The accuracy based on the distance of each image can be separated into three spaces: the images have the distance less than average value (4 pixels): **56.66**%; the images have the distance from average value to 10 pixels: **40.27**%; and the images have the distance greater than 10 pixels: **3.07**%. The network has enabled to detect the landmark on pronotum automatically.

Fig.10 shows the proportion of acceptable landmarks. In our case, a predicted landmark is acceptable if the distance between it and corresponding manual landmarks is less than the average distance plus a value of standard deviation. Most of the landmarks have been detected with the accuracy greater than 70%.

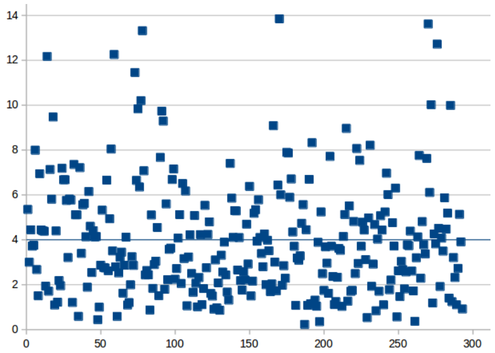


Fig. 9: The distribution of the distances on the first landmark. The blue line is the average value of all distances.

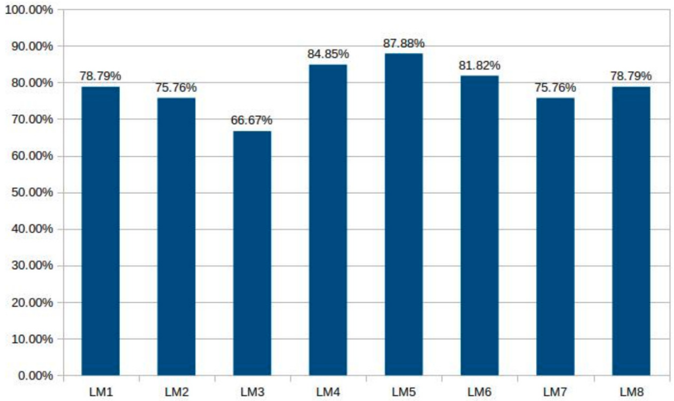


Fig. 10: The proportion of acceptable predicted landmarks

To have a comparison with the related works, we have calculated the same quality metrics for regression problems as Cintas et al. [9], *such as,* coefficient of determination (r^2), explained variance (EV), and Pearson correlation, using *scikit-learn* [26]. The best possible scores of all metrics are 1.0, lower values are worse. The comparing of the quality metrics with the previous works has been shown in Table.II.

TABLE II: The performance of the last architecture network on regressions quality metrics, comparing with the results of Cintas et al. [9]

Metric	$\mathbf{r^2}$	**EV**	**Pearson**
Cintast et al. [9]	0.884	0.951	0.976
Our	**0.9952**	**0.9951**	**0.9974**

At the test phase, the trained network is used to predict the landmarks on a set of test images. The program outputs the predicted-landmarks of the images as TPS files; in additional, it also fills and displays the predicted-landmarks on sixteenth firstly images of test data. With the outputs are TPS files, the user can use MAELab [2] framework[3] to display the landmarks on the images.

[2] An epoch is a single pass through the full training set.

[3] MAELab is a free software written in C++. It can be directly and freely obtained by request at the authors.

V. Conclusion and future works

In the context of collaboration with the biologist, the project towards replacing the manual landmarks by the automatic landmark on beetle's images. With beetle mandibles images, the object is easy to segment and we have succeeded to determine the landmarks automatically. In opposite, the pronotum images are difficult to segment. Methods which not suppose to be based on segmentation are necessary. In this paper, after testing several models, we have presented a convolutional neural network for automatic detection landmarks on the pronotum. It includes three times repeated structure which consists of a convolutional layer, a max pooling layer, and a dropout layer, followed by the connected layers. During the training phase, suitable techniques are used to prevent overfitting, a common issue of the neural networks. The network was trained several times in different selections of training data. After training with the manual landmarks given by the biologist, the network is able to predict the landmarks on the set of unseen images.

The results from the test set have been evaluated by calculating the distance between manual landmarks and corresponding predicted-landmarks. The average of distance errors on each landmark has been also considered. Using the convolutional network to predict the landmarks on biological images is promising good results in the case that the image can not be segmentation. The quality of prediction allows using automatic landmarking to replace manual landmarks in some aspects. In our case, the training dataset is limited. As a result, the accuracy of the network is acceptable. However, when we expect more about the accuracy of predicted landmarks (coordinates of predicted landmarks), the result of this work is still needed to improve (for example using a larger training dataset). Therefore, future research in landmarking identification appears as an improved of the worth exploring.

Acknowledgment

The research has been supported by DevMAP project[4]. We would like to thank my colleague, ALEXIA Marie, who has provided manual landmarks on beetle images.

References

[1] Y. LeCun, K. Kavukcuoglu, and C. Farabet, "Convolutional networks and applications in vision," in *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*. IEEE, 2010, pp. 253–256.

[2] V. L. LE, M. BEURTON-AIMAR, A. KRÄHENBÜHL, and N. PARISEY, "MAELab: a framework to automatize landmark estimation," in *WSCG 2017*, Plzen, Czech Republic, May 2017. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01571440

[3] J. Canny, "A computational approach to edge detection," *IEEE Transactions on pattern analysis and machine intelligence*, no. 6, pp. 679–698, 1986.

[4] J. Shlens, "A tutorial on principal component analysis," *arXiv preprint arXiv:1404.1100*, 2014.

[5] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[6] S. Palaniswamy, N. A. Thacker, and C. P. Klingenberg, "Automatic identification of landmarks in digital images," *IET Computer Vision*, vol. 4, no. 4, pp. 247–260, 2010.

[7] Y. Sun, X. Wang, and X. Tang, "Deep convolutional network cascade for facial point detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 3476–3483.

[8] Z. Zhang, P. Luo, C. C. Loy, and X. Tang, "Facial landmark detection by deep multi-task learning," in *European Conference on Computer Vision*. Springer, 2014, pp. 94–108.

[9] C. e. a. Cintas, "Automatic ear detection and feature extraction using geometric morphometrics and convolutional neural networks," *IET Biometrics*, vol. 6, no. 3, pp. 211–223, 2016.

[10] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[12] D. Ciregan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 3642–3649.

[13] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[14] T. Mikolov, A. Deoras, D. Povey, L. Burget, and J. Černockỳ, "Strategies for training large scale neural network language models," in *Automatic Speech Recognition and Understanding (ASRU), 2011 IEEE Workshop on*. IEEE, 2011, pp. 196–201.

[15] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[16] T. N. Sainath, A.-r. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for lvcsr," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 8614–8618.

[17] A. Bordes, S. Chopra, and J. Weston, "Question answering with subgraph embeddings," *arXiv preprint arXiv:1406.3676*, 2014.

[18] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[19] S. Jean, K. Cho, R. Memisevic, and Y. Bengio, "On using very large target vocabulary for neural machine translation," *arXiv preprint arXiv:1412.2007*, 2014.

[20] H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua, "A convolutional neural network cascade for face detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5325–5334.

[21] J. J. Tompson, A. Jain, Y. LeCun, and C. Bregler, "Joint training of a convolutional network and a graphical model for human pose estimation," in *Advances in neural information processing systems*, 2014, pp. 1799–1807.

[22] S. Dieleman, J. Schlter, C. Raffel, E. Olson, S. K. Snderby, D. Nouri *et al.*, "Lasagne: First release." Aug. 2015. [Online]. Available: http://dx.doi.org/10.5281/zenodo.27878

[23] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient backprop," in *Neural networks: Tricks of the trade*. Springer, 2012, pp. 9–48.

[24] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting." *Journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[25] J. M. Bland and D. G. Altman, "Statistics notes: measurement error," *Bmj*, vol. 313, no. 7059, p. 744, 1996.

[26] P. et al, "Scikit-learn: Machine learning in python," *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[4] https://www6.rennes.inra.fr/igepp_eng/Research-teams/Demecology/Projects/INRASPEDevMAP