Towards landmarks prediction with Deep Network

Van Linh Le[1,3], **Marie Beurton-Aimar**[1], **Akka Zemmari**[1], **Nicolas Parisey**[2]

[1]LaBRI - CNRS 5800 Bordeaux University, France, van-linh.le/beurton/zemmari@labri.fr
[2]IGEPP - INRA 1349, France, nparisey@rennes.inra.fr
[3]ITDLU - Dalat University, Vietnam, linhlv@dlu.edu.vn

Keywords: Landmarks, convolutional neural networks, fine-tuning, recogntion, procrustes.

Abstract

Morphometry landmarks are used in many biological applications. Mostly, the landmarks are defined manually or semi-automatic by applying the image processing techniques. In recent years, deep learning is known as a good solution for the difficult problems in computer vision. It appears in many fields such as classification, recognition, face detection. In the context applying deep learning to solve the regression problems, in this paper, we present a convolutional neural network to predict the landmarks on 2D images, specify beetle's pronotum images. The experiments on the proposed network have been done in two ways: training from scratch and fine-tuning from a trained model. The quality of predicted landmarks is evaluated by calculating the distance in pixels between the coordinates of the predicted landmarks and manual landmarks which were provided by the biologists. The dataset includes 293 images has been used to experiment the method.

1 Introduction

Morphometry landmark (or point of interest) is an important feature in many biological investigations. It was usually used to analyze the forms of whole biological organs or organisms. The analysis is mainly based on the coordinates of the landmarks. The collecting of enough the number of landmarks can help the biologists make a good estimate about organisms. Depending on the problem, the number of landmarks may be more or less; besides, the location of landmarks can be located on the shape (border) or inside the object, *for examples,* the landmarks on Drosophila wings have stayed on the veins of the wings but the landmarks on human ear can be located at the ear hole or inside. Recently, the landmarks were set manually by the biologists. This work is time-consuming and difficult to reproduce. Therefore, a method that proposes automatically the coordinates of landmarks could be a concern.

Based on the characteristics of digital images acquired for morphological studies, the images can be divided into two groups: the images where they are easy to segment the objects of interest, called *segment-able images*; and the images that we can go in tight when segment the objects, called *un-segment-able images*. For that reason, the methods that used to identify the landmarks automatically may be divided into two groups too. For segment-able images, identification of landmarks on the shape can be finished by applying the image processing techniques such as HOG[1], SIFT[2], ... But for un-segment-able images, defining the landmarks become a challenge and the image processing techniques seem to be inappropriate. This article introduces two scenarios for automatic detection of the landmarks on biological images, specific beetle's images. The dataset includes 293 images of beetles. For each beetle, five parts have been extracted (*pronotum, head, body, left and right mandible*) with their manual landmarks by the biologists. So, the coordinates of the manual landmarks can be seen as the ground truth to evaluated the predicted landmarks. In this article, a Convolutional Neural Network (CNN)[3] has been designed and used in two ways to predict the landmarks: in the first way, the network has been trained from scratch on the dataset of pronotum images. In the second way, the network has been trained on a "combination dataset" includes the images of three parts of beetle, firstly; then the trained model was used to fine-tune [4] on pronotum dataset. The output model uses to predict the landmarks on a test set of pronotum (Fig.1).

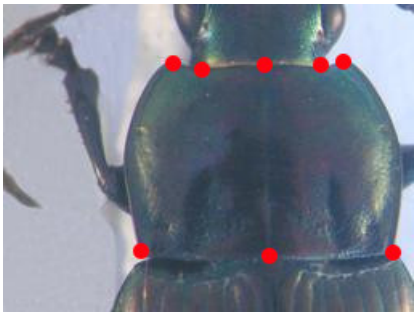


Figure 1: An example of pronotum images and its manual landmarks

In the next section, we present related works in domain automatically estimation landmarks on 2D images. In section 3, we present the architecture of the network and the procedure to generate the data images. Two scenarios to predict the landmarks by using the network are discussed in Section 4. In the last section, we have some conclusion based on the experiments and analyzing the results.

2 Related works

Deep learning methods are coming from machine learning theory. They were introduced in a long time ago but the development was still limited. Recently, the improvement of computing capacity, both in memory size and time with GPU programming has opened a new challenge for deep learning. Many deep learning architectures have been proposed to solve the problems of classification [5, 6], image recognition [7, 8, 9], speech recognition [10, 11] and language translation [12, 13]. Along with that development, many frameworks have been built such as Caffe [14], Theano [15], Tensorflow [16],.... The appearance of the frameworks has helped the users easier to work on deep learning. In image analysis field, deep learning, specifically CNN, is used to predict the key points on the image. Yi Sun et al. [17] have proposed a cascaded convolutional network to predict the key points on the human face. Zhang et al. [18] optimizes facial landmarks detection with a set of related tasks such as head pose estimation, age estimation, ... Cintas et al. [19] have introduced a network to predict the landmarks on human ear images to characterize ear shape.

In geometric morphometrics, landmarks or points of interest are one of the important characteristics. Landmark studies have traditionally analyzed on 2D images. Depending on which situation was stayed (segment-able or un-segment-able images), setting landmarks must apply the different methods.

When segmentation can be applied, Lowe et al. [2] have proposed a method to identify the key points in the 2D image. From the detected key points, the method is able to match two images. Palaniswamy et al. [1] have applied probabilistic Hough Transform to automatically estimate the landmarks in images of Drosophila wings. Krahenbuhl et al. [20] have extended Palaniswamy's method to detect landmarks automatically on beetles mandibles. Unfortunately, this method can not be applied to other parts of beetle that the segmentation has too many noises, such as pronotum images. Fortunately, the applying deep learning sees that a suitable solution to predict the landmarks on un-segment-able images.

3 Network model

Deep learning presents a learning method with multiple levels of representation of connected layers (convolutional neural network). Data representation is transformed from a lower level to a higher level with many complex functions can be learned via backpropagation. In this section, we present a CNN that we have used to predict the landmarks.

3.1 Network architecture

The first step to work in CNN is to study the network architecture. After several tests, we have chosen for this application to work with a model provided in Lasagne framework coming from Theano. In the first section, we will present the original model and then, we will describe how we have modified it by definition of an elementary block that we compose in the final model.

Like the networks [3, 9, 19], the proposed network consists of common layers with different learnable parameters. It receives an input of $1 \times 256 \times 192$ to train, to validate, and to test. The network consists of three repeated-structures of a convolutional layer followed by a maximum pooling layer. The depth of convolutional layers increases with different size of the filter kernels. All the kernels of pooling layers have the same size. At the end, three full connected layers have been added to the network. The output of the last full-connected layer corresponds to 8 landmarks (x and y coordinates) which we would like to predict. In general way, the layers and their parameters are presented as following:

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

. Table.1 shows the sequence layers in the model along with the parameters at each layer of the original model.

Order	Layer	Order	Layer
1	Input($1 \times 256 \times 192$)	6	CONV(128,2,1,0)
2	CONV(32,3,1,0)	7	POOL(2,2,0)
3	POOL(2,2,0)	8	FC(1000)
4	CONV(64,2,1,0)	9	FC(1000)
5	POOL(2,2,0)	10	FC(16)

Table 1: The layers and their parameters in the original model

The experiment on original model shows that this architecture is still not good enough to predict the landmarks. It is still general and overfitting has appeared during training and validation. To prevent the overfitting, four dropout layers have been added into 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 [21]. Fig.2 presents the final architecture of the model. The first three dropout layers are supplemented to the repeated-structures followed the maximum pooling layers. In that way, structure becomes an elementary block includes a convolution layer (C_i) followed by a maximum pooling (P_i) and dropout layer (D_i)(with $i = 1..3$). 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, pooling and full-connected layers as the previous architecture. The remaining dropout layer (D_4) is inserted between the first two full connected layers (FC_1 *and* FC_2). The probability value of this layer is set to 0.5. The output layer (FC_3) still contains 16 units corresponding to the coordinates of 8 predicted landmarks.

During training, the values of learnable parameters have been updated to increase the accuracy of the network by applying gradient descent in backward phase. Therefore, the network is designed with a small sharing learning rate and momentum. Their values are updated over training time to fit with

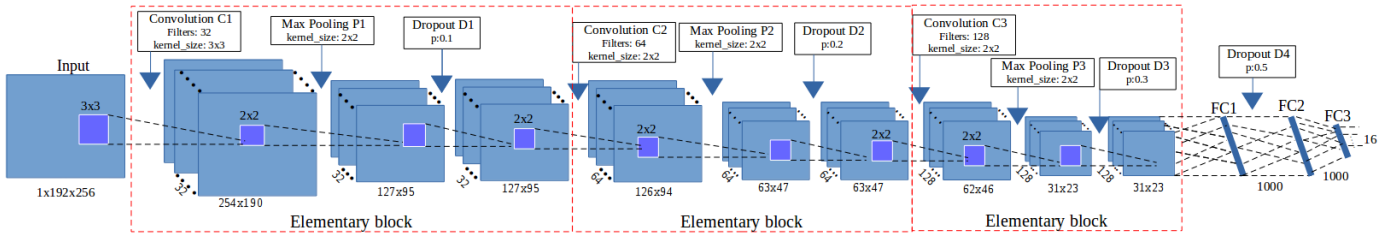


Figure 2: The last version of the proposed convolutional neural network

the number of epochs[1]. The network is designed to finish the training in 5000 epochs. The learning rate was initialized at 0.03 and stopped at 0.00001, while the momentum was updated from 0.9 to 0.9999. Because landmarks prediction can be seen as a regression problem in deep learning. Therefore, the root mean square error (RMSE) was used as a quality metric to evaluate the result and compute the losses of the proposed architecture. The implementation of the network is done on Lasagne framework [22] which allows computing on GPU. The network has been trained on NVIDIA TITAN X cards.

3.2 Enlarge the dataset

The dataset includes 293 images of beetles (for each part). All images are taken with the same camera in the same condition with a resolution of 3264×2448. Each image has been set 8 manual landmarks by biologists (Fig. 1). The dataset was split into two subsets: training (and validation) set contains 260 images and testing set includes 33 images. In most of CNNs [3, 17, 5, 19], the size of the input is limited to 256 pixels. In our case, the resolution of input image seems that too large and it becomes a difficulty for the network. So, the images are down-sampling to a new resolution 256×192 before training and testing. Of course, the coordinates of manual landmarks are also scaled to fit with the new resolution of images.

Besides the size of the input, the number of images is also a challenge when applying CNN. Normally, training a CNN with a large dataset will give us the result better than when we training CNN on a small dataset. Moreover, working with a small dataset, we can meet a popular problem, *overfitting*. So, we need to enlarge the size of the dataset instead of 260. In image processing, we usually apply transform procedures (translation, rotation) 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. To address this problem, we have applied two procedures to enlarge the size of the dataset.

The first procedure is applied to change the value of each channel in the original image. According to that, 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 of the procedures, 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).

3.3 First results

In this scenario, the network was trained on a dataset of 1820 pronotum images which were generated from 293 original pronotum images by applying the procedure in section 3.2. The number of images that used for training and validation is splitted randomly by a ratio (training: 60%, validation: 40%) that has been set during the network setup. During the training, the network learned the information through a pair of *(image, landmarks)* in training set. At the test phase, the image without landmarks was given to the trained network and the predicted landmarks will be given at the output. In practical of 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.

To obtain all the predicted landmarks for all pronotum images (instead of 33 images), a situation to choose the test images is executed, called *round*. For each round, a set of 33 images have been chosen for the test set; the remaining images have been put into the training set. Following that, the network will be trained with many different training datasets and the output model will be used to predict the landmarks on the images in the corresponding test set. Table.2 shows the losses during training the network on pronotum images.

Fig.3 shows the training error and the validation error during training time of one round. The blue curve presents RMSE error on training data. The green curve presents the validation error. Clearly, the losses are very different from the beginning. But, the difference is narrowed when the epoch increase.

Besides the losses during training, the accuracy on coordinates of predicted landmarks in the test images is also considered. Firstly, the trained model was used to predict the landmarks on all images in the test set. Then, the distances (in pix-

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

Round	Training loss	Validation loss
1	0.00018	0.00019
2	0.00019	0.00021
3	0.00019	0.00026
4	0.00021	0.00029
5	0.00021	0.00029
6	0.00019	0.00018
7	0.00018	0.00018
8	0.00018	0.00021
9	0.00020	0.00027

Table 2: The losses during training the model on pronotum images dataset

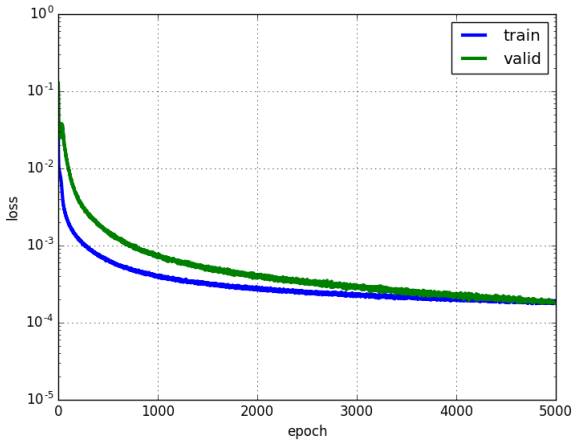


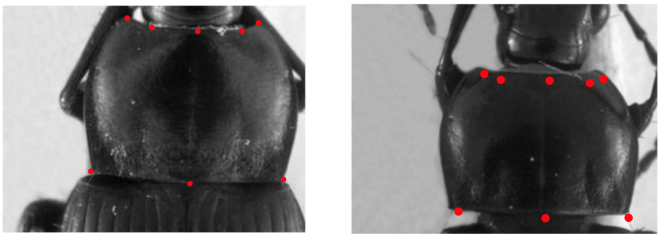
Figure 3: The loss curves during training of proposed network

els) between manual and corresponding predicted landmarks in each image were calculated as the error distances. Finally, the error distance per landmark was calculated for all test images. Table.3 shows the average error distance given on each landmark. With the size of the images is 256×192, if we accept an error around 3% of the image size (~ 3.5 pixels), the error distances are acceptable.

#Landmark	Distance
1	4.002
2	4.4831
3	4.2959
4	4.3865
5	4.2925
6	5.3631
7	4.636
8	4.9363

Table 3: The average error distance per landmark

Fig.4 shows the predicted landmarks on two test images. When we consider the accuracy of predicted landmarks by calculating the distance between manual and corresponding predicted landmarks, the accuracy on coordinates of predicted landmarks on Fig.4a is 99% and the propotion on Fig.4b is 80%.



(a) Image with well-predicted landmarks

(b) Image with inaccuracy landmarks

Figure 4: The predicted landmarks on the images in test set. The red points present for the predicted landmarks

As a result, the network is able to predict the landmarks on a test set of pronotum images. At statistic side, the predicted landmarks are acceptable. But in image processing side, we expect more about the accuracy (coordinates of predicted landmarks), and the result of CNN is still needed to improve.

4 Fine-tuning to transfer learning

In section 3.3, the proposed network has been experimented by training from scratch on pronotum dataset. The results of experiments have shown that the network has worked well to detect the landmarks on the pronotum images. However, when we consider the predicted landmarks by displaying the landmarks on the images, the result is still not precise, the average error still high (≥ 4 pixels).

In order to reach more acceptable results for biologists, we have broadened the model with the step of transfer learning. That is a method that we re-use the model developed for a task on another task. It allows rapid progress or improved the performance of the model on the second task [24]. In our case, we have applied fine-tuning, a strategy of transfer learning. Fine-tuning is to not only replace and retrain the model on the new dataset, it also fine-tunes the weight of trained model by continuing the backpropagation. So, instead of training and testing on pronotum images dataset, the network will be trained on the dataset includes the images of three parts of beetle i.e pronotum, body and head. Then, the trained model will be used to fine-tune and test on pronotum set.

4.1 Training data preparation

The training dataset includes a combination of the images from three sets: pronotum, body and head (Fig.5). For each set, 260 original images have been chosen radomly for training and validation. By applying the same procedure in section 3.2, the training dataset was enlarged to 5460 images ($260 \times 7 \times 3$). However, the number of manual landmarks on each part is difference: 8 *landmarks on pronotum part,* 11 *landmarks on body part, and* 10 *landmarks on head part*. The manual landmarks have a specific meaning for the biologists. So, we can not insert the landmarks arbitrarily. Instead of, we will keep the smallest

number of landmarks among three parts and we remove some landmarks on other parts. Therefore, we kept the number of the landmark on pronotum as a reference and we suppressed some landmarks on the body and head part. Specifically, we have removed three landmarks on the body part ($1^{st}; 6^th; 9^{th}$) and two landmarks on the head part ($5th; 6^{th}$) (Fig.5).

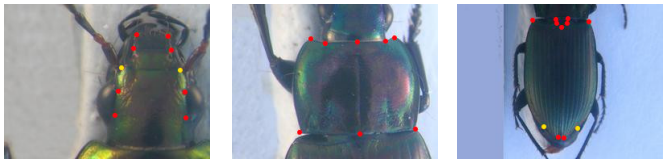


Figure 5: The dataset images. The *red* points represent the landmarks that are kept, while *yellow* points represent for the suppressed landmarks during training the model

4.2 Using fine-tuning for pronotum dataset

The training data includes 5460 images was trained on the proposed network (section 3.1) with the same parameters than we have trained on pronotum images. After that, the trained model have been continued to fine-tune on pronotum dataset. To compare the result with the previous one, the trained model has been fine-tuned in many rounds with different datasets. The losses during fine-tuning are shown in Table.4. Comparing with the losses when we trained the model from scratch (Table. 2), the validation losses of this scenario are significantly improved (around 40%).

Round	Training loss	Validation loss
1	0.00019	0.00009
2	0.00018	0.00010
3	0.00018	0.00010
4	0.00019	0.00008
5	0.00019	0.00009
6	0.00018	0.00008
7	0.00019	0.00008
8	0.00018	0.00006
9	0.00018	0.00009

Table 4: The losses during fine-tuning model

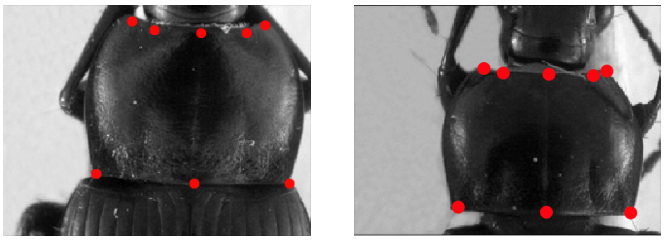
The output model has been used to predict the landmarks on the test images. Then, the average errors based on the distances between predicted and corresponding manual landmarks are given. The results are shown in Table.5. The **Error 1** column presents for the average error during training from scratch on pronotum images; the **Error 2** column presents for the average error during fine-tuning the pronotum from the trained model; and the **Percentage** column presents for the difference between the distance errors in percentage. Clearly, when we compare the average errors between two scenarios, the result of predicted landmarks in the second scenario is more precise than the first one.

Fig.6 shows the predicted landmarks by applying the second scenario on the same test images as Fig.4. The red points

#Landmark	Error 1	Error 2	Improved(%)
1	**4.002**	**2.486**	**37.88**
2	4.4831	2.720	39.33
3	4.2959	2.652	38.27
4	4.3865	2.771	36.83
5	4.2925	2.487	42.06
6	**5.3631**	**3.049**	**43.15**
7	4.636	2.684	42.11
8	4.9363	2.871	41.84

Table 5: The average error distance per landmark.

present for the predicted landmarks. In Fig.6a, the positions of predicted landmarks are the same when we compare with the result from Fig.4a; but in Fig.6b, the coordinates of predicted landmarks are strongly improved.



(a) Image 1 (b) Image 2

Figure 6: The predicted landmarks on the images in test set with fine-tuning model.

As a result of working, the program outputs the predicted-landmarks of the images as TPS files. With the outputs are TPS files, the user can use MAELab framework[2] to display the landmarks on the images.

5 Conclusion

In this paper, we have presented a CNN with two scenarios to predict the landmarks on beetle's pronotum images. The CNN network has been designed with three times repeated structure which consists of a convolutional layer, a max pooling layer, and a dropout layer, followed by the connected layers. In the training phase, the CNN have been trained with several times in different selections of training data. After training, the network was able to predict the landmarks on the images in the test set.

In the first scenario, the model has been trained (from scratch) and tested on the dataset of pronotum images. While in the second scenario, the model has been trained on a dataset includes the images from three parts of beetles. Then, the trained model has been used to fine-tune and test on pronotum images.

The result has been evaluated by comparing the coordinates between predicted and manual landmarks. The results have shown that using the convolutional network to predict the landmarks on biological images is promising good results in

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

the case that the image was difficult to segment. The quality of prediction allows using automatic landmarking to replace manual landmarks in some aspects. Training model from scratch or fine-tuning the trained model are given the acceptable results. But with a limited number of data, we need to improve the results a little bit. Therefore, future research in landmarking identification appears as an improved of the worth exploring.

Acknowledgements

The research has been supported by DevMap project. We would like to thank the my colleague, ALEXIA Marie, who have provided manual landmarks on beetle images.

References

[1] 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.

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

[3] 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*, pp. 253–256, IEEE, 2010.

[4] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?," in *Advances in neural information processing systems*, pp. 3320–3328, 2014.

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

[6] 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*, pp. 3642–3649, IEEE, 2012.

[7] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, *et al.*, "Going deeper with convolutions," Cvpr, 2015.

[8] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, "Learning hierarchical features for scene labeling," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1915–1929, 2013.

[9] 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*, pp. 5325–5334, 2015.

[10] 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*, pp. 196–201, IEEE, 2011.

[11] 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.

[12] 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.

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

[14] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.

[15] Theano Development Team, "Theano: A Python framework for fast computation of mathematical expressions," *arXiv e-prints*, vol. abs/1605.02688, May 2016.

[16] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

[17] 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*, pp. 3476–3483, 2013.

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

[19] C. Cintas, M. Quinto-Sánchez, V. Acuña, C. Paschetta, S. de Azevedo, C. C. S. de Cerqueira, V. Ramallo, C. Gallo, G. Poletti, M. C. Bortolini, *et al.*, "Automatic ear detection and feature extraction using geometric morphometrics and convolutional neural networks," *IET Biometrics*, vol. 6, no. 3, pp. 211–223, 2016.

[20] V. L. Le, M. Beurton-Aimar, A. Krahenbuhl, and N. Parisey, "MAELab: a framework to automatize landmark estimation," in *WSCG 2017*, 25th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision'2017, (Plzen, Czech Republic), May 2017.

[21] 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.

[22] S. Dieleman, J. Schlter, C. Raffel, E. Olson, S. K. Snderby, D. Nouri, *et al.*, "Lasagne: First release.," Aug. 2015.

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

[24] L. Torrey and J. Shavlik, "Transfer learning," *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques*, vol. 1, p. 242, 2009.