Effective Training of Convolutional Neural Networks for Insect Image Recognition

Maxime Martineau[1], Romain Raveaux[1], Clément Chatelain[2], Donatello Conte[1], and Gilles Venturini[1]

[1] LIFAT EA 6300, 37200 Tours
[2] LITIS EA 4108, 76800 Saint-Etienne du Rouvray, France

Abstract. Insects are living beings whose utility is critical in life sciences. They enable biologists obtaining knowledge on natural landscapes (for example on their health). Nevertheless, insect identification is time-consuming and requires experienced workforce. To ease this task, we propose to turn it into an image-based pattern recognition problem by recognizing the insect from a photo. In this paper state-of-art deep convolutional architectures are used to tackle this problem. However, a limitation to the use of deep CNNs is the lack of data and the discrepancies in classes cardinality. To deal with such limitations, transfer learning is used to apply knowledge learnt from ImageNet-1000 recognition task to insect image recognition task. A question arises from transfer-learning : is it relevant to retrain the entire network or is it better not to modify some layers weights? The hypothesis behind this question is that there must be part of the network which contains generic (problem-independent) knowledge and the other one contains problem-specific knowledge. Tests have been conducted on two different insect image datasets. VGG-16 models were adapted to be more easily learnt. VGG-16 models were trained a) from scratch b) from ImageNet-1000. An advanced study was led on one of the datasets in which the influences on performance of two parameters were investigated: 1) The amount of learning data 2) The number of layers to be finetuned. It was determined VGG-16 last block is enough to be relearnt. We have made the code of our experiment as well as the script for generating an annotated insect dataset from ImageNet publicly available.

1 Introduction

Insects are a class of invertebrates within the arthropods that have an exoskeleton, a three-part body (head, thorax and abdomen), three pairs of jointed legs, compound eyes and one pair of antennae. With 1,5 millions of species, they are more representative for wholesale organism biodiversity than any other group. Arthropods have been recognized as efficient indicators of ecosystem function and recommended for use in conservation planning [16]. Building accurate knowledge of the identity, the geographic distribution and the evolution of insect species is essential for a sustainable development of humanity as well as for biodiversity conservation. Finding automatic methods for such identification is

an important topic with many expectations. One of the most common data that can be used in this context are images. Images of arthropods can be acquired and further processed by an image classification system. The literature about insect image captures can be split into two broad categories: lab-based setting and field-based setting. In a lab-based setting there is a fixed protocol for image acquisition. This protocol governs the insect trapping, its placement and the material used for the acquisition (capture sensor, lighting system, etc.). Lab-based setting is mainly used by entomologists bringing the insects to the lab to inspect them and to identify them. Field-based settings means insect images taken directly in cultivated fields, without any particular constraints to the image capture system. An intermediate image type is multi-individuals which show many individuals at the same time, in a lab-based environment. Deep neural networks (DNN) have been extensively used in pattern recognition and have outperformed the conventional methods for specific tasks such as segmentation, classification and detection. However, to our knowledge, DNN have never been applied to insect image recognition [14].

Considering the hierarchical feature learning fashion in DNN, first layers are expected [20] to learn features for general simple visual building blocks, such as edges, corners and simple blob-like structures, while the last layers learn more complicated abstract task-dependent features. In general, the ability to learn domain-dependent high-level representations is an advantage enabling DNNs to achieve great recognition capabilities. However, DNN require large labeled dataset to efficiently learn their huge number of parameters, and insect image databases only contains a few hundred of labeled samples. To overcome this lack of data, transfer learning has been proposed which aims at transferring the knowledge from a more or less related task that has already been learned on a large dataset. Although transfer learning has been shown to be very efficient on many applications, its limits and its practical implementation issues has not been much studied. For example, it would be practically important to determine how much data on the target domain is required for domain adaptation with sufficient accuracy for a given task, or how many layers from a model fitted on the source domain can be effectively transferred to the target domain. Or more interestingly, given a number of available samples on the target domain, what layer types and how many of those can we afford to learn. Moreover, there is a common scenario in which a large set of annotated data is available (ImageNet-arthropod subset), often collected in a time-consuming and costly process. To what extent these data can contribute to a better analysis of new datasets is another question worth investigating.

In this study, we aim towards answering the questions discussed above. To tackle the insect image recognition problem, we use transfer learning methodology for domain adaptation of models trained on scene images. The contribution of this paper is an effective learning methodology for the insect recognition problem. The role of pretraining in the DNN accuracy is detailed and the question is rised about how many insect samples are required to achieve a sufficient accuracy from the biologist viewpoint.

2 Problem definition : Image-based arthropod classification

This section introduces the problem tackled in this article: Image-based arthropod classification.

Image-based arthropod classification could be seen as an application of image classification. Based on some photograph depicting the specimen, its biological identity is to be determined. The peculiarities of the problem are three-fold : rarity of images, image variations and large discrepancies among class cardinalities.

Rarity of images. Only experts such as taxonomists and trained technicians can identify accurately insect classes, because it requires special skills acquired through extensive experience. In lab-based setting, most of the acquisition systems are manually manipulated which increase the workforce amount (see examples on Figure 1(a) and Figure 1(b)). **Image variations**. Aside from classical object image variations (such as rotation, scale, parallax, background or lighting), insect images have more particular properties such as pose (because specimen appearance varies with the orientation they are been shown) and deformation (because the specimens are most of the time composed of articulated parts). These aforementioned variations can be referred to as *capture-specific* variations in the sense they only depend on capture factors. About the objects themselves (*object-specific* variations), age, sex and genetic mutations are the main factors of visual variations. The most instructive example is that of lepidoptera (commonly referred to as butterflies) which can have extremely different visual aspects along time, being successively caterpillars, chrysalises and butterflies. **Large discrepancies among class cardinalities**. Insect capture campaigns are season-dependent impacting the number and the type of the captured insects. This fact can be translated in the pattern recognition domain as an imbalanced classification problem.

3 State of the art

Image-based insect recognition is not a newly addressed problem. A detailed study was lead [14]. This study focused successively on capture protocols, feature extraction methods and finally classification. The last two points are of first interest here while the first point is to be considered as an input constraint and depends fully on the biologists workflow. Moreover, the datasets and the classes that constitute them are very different and motivated by different biological scopes and problems.

(a) (b)

Fig. 1: Lab-based samples.

Regarding feature extraction, the first pieces of work tackle the problem in very ad-hoc ways. [8] extract dimensions on the insects wings using venations as keypoints. Others [17] use geometrical attributes from the region of interest as features. Then, studies began to emerge using standard local and global features such as SIFT [18]. On top of these standard handcrafted features are learnt higher level features using MLPs [1], Bag of Words [12] or Sparse Coding [19]. The next step is to introduce hierarchically learnt descriptors, with many levels of abstraction. [18] uses a stack of Denoising Autoencoders to this extent. The observation that can be done is the features are no more particular to the problem itself and is about learning hierarchical representations to get satisfying feature spaces that suit the biologists goals through learning on their images. In such a frame, [1] applies MLPs on raw pixel even though the problem is a very simple one (two classes : harmful/harmless insects).

Deep neural networks are machine learning models that are now the state-of-the-art models for almost every pattern recognition tasks. Their main strength is their ability to learn suitable features for a given task, thus avoiding the need for handcrafted features design. Convolutional Neural Networks (CNN) are an instantiation of DNN dedicated to image processing. Through a mechanism of shared weights, their input layers are convolutional filters that automatically learn features from multidimensional signals [10]. CNN now outperform most of traditional approaches (e.g. based on handcrafted features) in various image analysis and recognition domains, such as natural scene recognition [10], Medical imaging [3], Image segmentation [11] or handwriting recognition [15]. However, although very efficient, deep CNN models require a huge amount of labeled data to avoid overfitting, which can be compared to "learning by heart" the training database. In order to circumvent these issues, the architectures are generally trained using many tricks such as regularization, dropout or data augmentation for improving their generalization ability. But all these hints do not replace a reasonable amount of labeled data for training the architectures.

Recently, the transfer learning idea has been proposed to train huge models with small labeled datasets. It is based on the exploitation of pre-trained models on a huge datasets from another domain. The model is then fine tuned on a specific, smaller database of the domain considered. Models are often trained on imagenet, a natural scene database of more than 14M images[10]. Even if the specific domain strongly differs from the imagenet database, the transfer learning have shown very impressive results on many tasks such as handwriting recognition [7], signature identification [9], and medical imaging [2].

4 Proposed approach

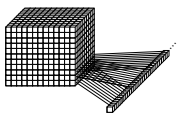
This article presents a method to easily train a deep convolutional neural network architecture using transfer learning, with application to the insect image recognition problem. The deep architecture has been adapted to be easily trained on low volume datasets and the feeding, learning and hyper-parameter optimization procedures are detailed in the next subsections.

4.1 Transfer learning adapted VGG-16 architecture

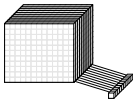
The fine-tuning CNN experiments involve VGG-16 instances pretrained on ImageNet 1000 [5]. Although more recent models exist (for example GoogLeNet or ResNet), preliminary experiments shown these architectures yielded similar results on the problem tackled here. Moreover, VGG-16 was chosen for its simplicity and relatively small number of layers.

VGG-16 has undergone a crucial modification in its convolutional end. The original model end (see Figure 3 on the left) consists of a three-layered MLP (FC1-3) which takes as inputs every coefficients from the last 2D activation map (block5-pool): The $7 \times 7 \times 512$ volume is flattened to obtain 25088 input values for the MLP. This has been replaced by a global average pooling filter which keeps only the averages of each of the 512 7×7-slices of the $7 \times 7 \times 512$ volume [13]. This reduces the input size of the MLP from 25088 to 512. A visualization of such a transformation is pictured in Figure 2. This modification acts as a regularization that limits overfitting as it reduces the number of neurons in the first part of the MLP, and takes more advantage to the structure of the last convolutional feature maps [13]. The remains of the MLP consist of a single fully-connected layer with 256 neurons and the prediction layer.

The overall transfer learning architecture is described in Fig. 3.



(a) A flatten layer



(b) A global pooling layer

Fig. 2: Layer translating volumes into 1-D vector to take as input of the fully-connected end of a CNN

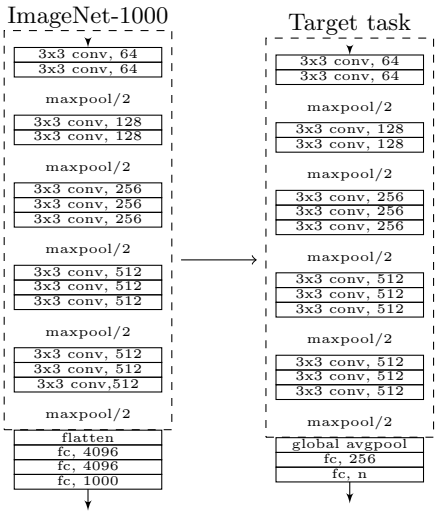


Fig. 3: Transfer learning architecture with modified VGG-16

4.2 Data preprocessing

Preparing the data to feed the neural network is critical to the learning process. It consists of two steps. The first one is about adapting the input image (size $n \times m \times 3$, n an m being respectively the height and width of the input image) to the receptive field of the neural network (size $224 \times 224 \times 3$, the size of the ImageNet input on the three color channels). The most straight-forward approach to this size adaption is to resize the image. The pros of this method lies in the fact that all the information of the image is fed to the network. The main cons is that the ratio of the image is not preserved. In the case of one dataset in use in this article, the images ratios are significantly various and thus resizing the images would end up in bad performance. The method used here consists in cropping the images in their center so that the image is of size $k \times k \times 3$ with $k = \min(m, n)$. Cropping instead of resizing keeps the image ratio as is. Finally, a resize operation can be performed from size $k \times k \times 3$ to size $224 \times 224 \times 3$. Once the size normalization is done, the images must be standardized across their dataset featurewise by subtracting the mean image and dividing by the standard deviation image. This is done separately for each color channel. Using the image unstandardized results in inability to train the system (the loss stagnating at its initial level). This behavior can be explained in the following manner: if the image is not standardized, the differences in scales causes the gradient descend to be applied at these different scales thus differently depending on the location or pixel value. Finally, data augmentation was used. The training images, each time they were fed to the network, were modified using a set of randomly chosen transformations (sheer, flip and zoom). It virtually augments the size of the set as many times as the number of epochs.

4.3 Learning rule

The data that is being dealt with is not only in small amount but also unequally distributed among classes. Neural networks are known to be very sensitive to cardinality discrepancies. There are two main solutions to address this problem. The first one consists of sampling on low-cardinality classes and the latter is to apply the gradient descent more or less depending on the cardinality of the examples classes. This latter method is the one being used here for the sake of simplicity. For each class $c \in Y$, the following weights have been applied: $w_c = \frac{\max_{c' \in Y} \text{Card}(c')}{\text{Card}(c)}$ where $\text{Card}(c)$ is the number of observations that fall into class c. w_c are used during the backpropagation step as weights to the error gradient descent term. The higher w_c, the stronger the error gradient is descended. The gradient descent method is vanilla Stochastic Gradient Descent with 0.9 Nesterov momentum [4]. On the contrary, the learning rate is divided by two if the loss hasn't been reduced in the span of 10 epochs. This intensification process enables the learning rate to be adapted depending on the scale of the gradient and therefore to avoid oscillations onto the search space. This behaviour is likely to lead to a local minimum. However in the case of multi-layer networks it is not

a major issue. [6] not only states this but also that global minima are prone to lead to overfitting in practice.

4.4 Hyper-parameter optimization

The main limitation to the use of deep models is the hyper optimization process it requires in order to get satisfying results. Additionally, using transfer learning during the training phase adds a new hyper-parameter: the number of layers to be trained on the target task. Deep CNN are composed of low level features that are independent from the problem. These low level features are located in the first layers. An efficient learning methodology should allow not spending time optimizing these layers. In our proposal, we show that only the last convolutional block (block5) and obviously the last dense layer which has ot been pretrained (fc1) are of first interest to train at ease the model while achieving a good classification rate. Conceptually speaking, block5 represents the high level features that are insect-dependent, while fc1 is a decision phase to fit at best the classification task in term of number of classes and class distribution. Reducing the search space as much as possible is therefore critical when learning with small amount of data.

5 Experimental work

This section details the lead experiments. Data in use is presented and both experiments and their respective results are described as well as their interpretations.

5.1 Datasets

The target domain is that of insect images and the task is to recognize the class they have been labeled in. Two sets have been used to that extent. The first set (IRBI) is a lab-based insect images set. Insects are captured into soapy liquid traps put at soil level. The captured specimens are then conserved in an alcohol solution before being identified. These same specimens are laid down on a plain background and under a constant and controlled lighting. Multiple shots are taken for a single individual : they are taken along 3 different orientation of the individual and with 7 different smartphones which makes up 21 shots for a single insect. The dataset split into train/valid/test sets has been performed at the insect level, in order to prevent the presence of an insect in two (or more) sets with different orientations, which would make the evaluation unfair. Besides, due to the cardinalities discrepancies, a stratified split method must be used at the class level. To have an analogous dataset with pictures taken in field-based settings, a subset of ImageNet was extracted: each leaf-synsets under the synset "arthropods" was used which make up 501 classes. In order to emulate the same constraints as in a real entomology set, the average class cardinality was lowered down to the average class cardinality of the IRBI set (see Tab.1). The statistical

validation method in use in this study is a 5-fold stratified cross-validation. To be able to apply such a split while keeping individuals from every class in the two sub-sets, classes with less than 5 individuals were removed.

Table 1: The image datasets

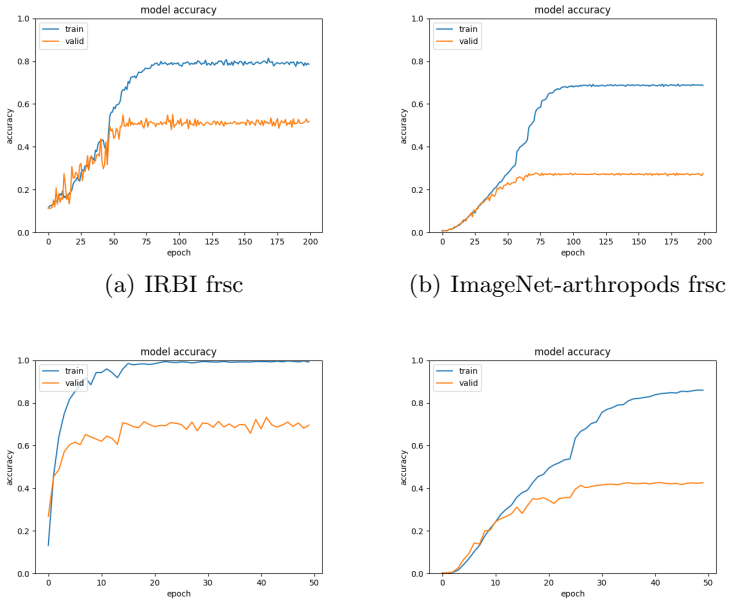
Dataset	Nb of classes	$\lvert\text{Card}(c)\rvert$	$\sigma(\text{Card}(c))$	$\min(\text{Card}(c))$	$\max(\text{Card}(c))$
IRBI	30	85	71	33	370
ImageNet-arthropods	443	96	78	6	392

5.2 Experiments

The first part of the experimental study is about comparing together i) a traditional method based on handcrafted features, ii) CNN-based models trained from scratch, and iii) these same models which exploit the transfer learning trick. Table 2 shows results on both image datasets and for the following models : **1) SIFTBoW :** a SIFT codebook representation combined with a SVM classifier with an RBF kernel. The SVM is optimized with a grid-search approach on the validation dataset for optimizing the penalty parameter ($C \in \{10^0, \ldots, 10^6\}$) and the kernel parameter ($\gamma \in \{10^{-6}, \ldots, 10^0\}$). **2) Several VGG-16 based CNN classifiers**, as described in the previous section : **a) *frsc*:** learnt from scratch with initial random weights (during 200 epochs); **b) *fitu*:** pre-trained network on ImageNet-1000 (during 50 epochs), and fine tuned on IRBI/Arthropods; **c) *fitu/w*:** same as **fitu**, with example weighting based on each class cardinality (see section 4.3). **d) *fitu7/w*:** same as **fitu/w** but with only 7 layers fine-tuned. All the code in use in this article can be retrieved here: `https://github.com/prafiny/deep-insect-experiments/`. A script to recreate the ImageNet-arthropods set is provided. In Table 2, the columnn "mean time" refers to the average time in seconds by epoch during the training. Additionally, the training curves (representing the recognition rate on validation set at each epoch) for VGG-16 both frsc and fitu/w are shown on Figure 4.

Several observations can be made on these experiments results. First, the traditional approach based on handcrafted features (SIFTBow) performs moderately. Second, the VGG-16 architecture trained from scratch (VGG16-frsc) on the target dataset slightly outperforms the traditional BoW approach. On the other hand, the VGG-16 fine-tuned with transfer learning (VGG-16-fitu) gives significantly higher results compared to those of the traditional approach Additionally, weighting training examples based on class cardinalities (VGG16-fitu/w) made the model better on average regarding the top-1 figures.

In a second time, an advanced study was conducted on the behavior and efficiency of transfer learning and CNN. In this study two factors are investigated:



(a) IRBI frsc (b) ImageNet-arthropods frsc

(c) IRBI fitu/w (d) ImageNet-arthropods fitu/w

Fig. 4: Learning curves for VGG-16 (blue for train accuracy, orange for valid accuracy).

the number of learning examples and the number of layer to be learnt. The results for one fold are shown in Figure 5. The dataset used in this frame is IRBI. While the effect of changing the number of training example is not so surprising, it is interesting to note that learning only on the last 7 layers seems enough to reach a 70 % accuracy score. This observation can be interpreted as follows: the ImageNet-1000 features are generic enough (up to the $22-7=15$th layer) to learn on the lab-based insect identification problem. Learning only the 7 last layers is in fact learning the last VGG16 block (block5) and the two dense layers. Also, it is equivalent to learn about half of the network weights. This experiment corroborates the statement made in Section 1 that the first layers

Table 2: Recognition rates on 5-fold experiments and mean epoch times

Model	**IRBI**			**ImageNet-arthropods**		
	Top-1	Top-5	mean time (s)	Top-1	Top-5	mean time (s)
SIFTBoW	52.3 % ± 3.7	82.7 % ± 3.3	—	11.7 % ± 0.2	25.9 % ± 0.4	—
VGG16-frsc	54.0 % ± 5.0	84.9 % ± 3.0	101	26.9 % ± 0.7	50.1 % ± 0.7	1470
VGG16-fitu	72.0 % ± 3.2	92.1 % ± 1.1	104.4	42.7 % ± 0.9	69.4 % ± 0.6	1473.6
VGG16-fitu/w	73.6 % ± 1.8	92.4 % ± 2.2	102.6	43.5 % ± 1.1	71.3 % ± 0.8	1473.2
VGG16-fitu7/w	72.4 % ± 2.8	92.6 % ± 2.1	52.4	43.3 % ± 0.6	71.8 % ± 0.4	721.6

can be considered as a generic feature extractor while the last convolutional block and the dense layer represents the task-dependent features. Last line of Table 2 shows results learning only 7 layers for both IRBI and ImageNet. Last but not the least, the training phase is sped up by a factor two by learning only the 7 last layers instead of the whole network.

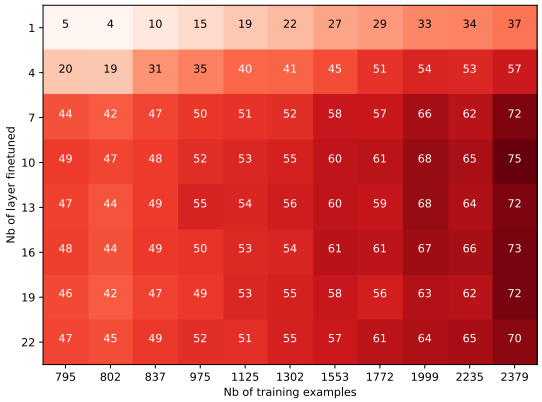


Fig. 5: Recognition rates on the testing set depending on the number of training examples and the number of layer finetuned

5.3 Misclassification analysis

Finally, some observations were made at the query level to have an insight on what are the possible causes of misclassification. The problem is that it is very tedious to get an idea of how CNNs discriminate between classes. In [21] two main methods classes are listed : 1) to display the convolution filters, 2) to use the filter responses as a tool to generate images. The drawback of such a method is that the produced images are often abstract and therefore hard to interpret. A possible solution to the interpretation issue is to rely on the database images rather than trying to make up images. In this extent, we adopted the following method: for each misclassified image, we retrieve the output vector of the CNN and try to find the images whose output vectors are nearest in the sense of vector space distance. This enables to work in a Content-Based Image Retrieval-like framework and listing the k-nearest neighbors images for a given query. Examples of query results produced by using this method are shown on Figure 6. Figure 6(a) shows a misclassification occurrence which is likely to be caused by the small apparent size of the insect. Each of the nearest images were from different classes yet quite resembling due to the low resolution of the capture. Figure 6(b) shows an example of a specimen mimicking another insect: syrphid flies colors enable them to be confused with bees or wasps suggesting they could sting a predator.

(a) Small specimen confused with other small specimens

(b) Resembling classes

Fig. 6: Misclassified samples. Left big image is the query image and right small images are the 5-nearest neighbors.

6 Conclusion

This study uses fine-tuned deep convolutional neural networks on unbalanced and low-volume image datasets. Transfer learning definitely helps to compensate the low amount of data as performances clearly outperform traditional approach when using the transfer learning trick. Also, we have found that training only the last convolutional block of VGG16 is sufficient to find an almost optimal solution. This can lead to the conclusion that ImageNet-1000 features are generic enough up to a high level of abstraction. Finally, weighting the examples to tackle the unbalanced learning problem did influence positively the results but only marginally.

7 Acknowledgments

This work was supported by a research grant from the Région Centre-Val de Loire, France.

References

1. S. M. Al-Saqer and G. M. Hassan. Artificial neural networks based red palm weevil (Rynchophorus Ferrugineous, Olivier) recognition system. *Am. J. Agric. Biol. Sci*, 6:356–364, 2011.
2. Y. Bar, I. Diamant, L. Wolf, and H. Greenspan. Deep learning with non-medical training used for chest pathology identification. *Proc. SPIE, Medical Imaging: Computer-Aided Diagnosis*, 9414:94140V–7, 2015.
3. S. Belharbi, C. Chatelain, R. Hérault, S. Adam, S. Thureau, M. Chastan, and R. Modzelewski. Spotting L3 slice in CT scans using deep convolutional network and transfer learning. *Computers in Biology and Medicine*, 87:95–103, 2017.

4. Y. Bengio, N. Boulanger-Lewandowski, and R. Pascanu. Advances in optimizing recurrent networks. *CoRR*, abs/1212.0901, 2012.
5. F. Chollet et al. Keras. `https://github.com/fchollet/keras`, 2015.
6. A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, and Y. LeCun. The loss surfaces of multilayer networks. In *Artificial Intelligence and Statistics*, pages 192–204, 2015.
7. D. C. Cireşan, U. Meier, and J. Schmidhuber. Transfer learning for latin and chinese characters with deep neural networks. In *Neural Networks (IJCNN), The 2012 International Joint Conference on*, pages 1–6. IEEE, 2012.
8. C. H. Dietrich and C. D. Pooley. Automated identification of leafhoppers (Homoptera: Cicadellidae: Draeculacephala Ball). *Annals of the Entomological Society of America*, 87(4):412–423, 1994.
9. L. G. Hafemann, R. Sabourin, and L. S. Oliveira. Writer-independent feature learning for offline signature verification using deep convolutional neural networks. *CoRR*, abs/1604.00974, 2016.
10. A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *NIPS 25*, pages 1097–1105. 2012.
11. M. Lai. Deep learning for medical image segmentation. *CoRR*, abs/1505.02000, 2015.
12. N. Larios, H. Deng, W. Zhang, M. Sarpola, J. Yuen, R. Paasch, A. Moldenke, D. A. Lytle, S. R. Correa, E. N. Mortensen, L. G. Shapiro, and T. G. Dietterich. Automated insect identification through concatenated histograms of local appearance features: feature vector generation and region detection for deformable objects. *Machine Vision and Applications*, 19(2):105–123, Mar. 2008.
13. M. Lin, Q. Chen, and S. Yan. Network in network. *CoRR*, abs/1312.4400, 2013.
14. M. Martineau, D. Conte, R. Raveaux, I. Arnault, D. Munier, and G. Venturini. A survey on image-based insect classification. *Pattern Recognition*, 65:273–284, 2017.
15. A. Poznanski and L. Wolf. Cnn-n-gram for handwriting word recognition. In *CVPR*, pages 2305–2314, 2016.
16. N. M. Van Straalen. Evaluation of bioindicator systems derived from soil arthropod communities. *Applied Soil Ecology*, 9(1):429–437, 1998.
17. J. Wang, C. Lin, L. Ji, and A. Liang. A new automatic identification system of insect images at the order level. *Knowledge-Based Systems*, 33:102–110, Sept. 2012.
18. C. Wen, D. Wu, H. Hu, and W. Pan. Pose estimation-dependent identification method for field moth images using deep learning architecture. *Biosystems Engineering*, 136:117–128, Aug. 2015.
19. C. Xie, J. Zhang, R. Li, J. Li, P. Hong, J. Xia, and P. Chen. Automatic classification for field crop insects via multiple-task sparse representation and multiple-kernel learning. *Computers and Electronics in Agriculture*, 119:123–132, Nov. 2015.
20. J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.
21. J. Yosinski, J. Clune, A. M. Nguyen, T. J. Fuchs, and H. Lipson. Understanding neural networks through deep visualization. *CoRR*, abs/1506.06579, 2015.