Fast Learning for Accurate Object Recognition Using a Pre-trained Deep Neural Network

Víctor Lobato-Ríos(✉), Ana C. Tenorio-Gonzalez, and Eduardo F. Morales

Instituto Nacional de Astrofísica, Óptica y Electrónica, Tonantzintla, Mexico
{vlobato,catanace17,emorales}@inaoep.mx

Abstract. Object recognition is a relevant task for many areas and, in particular, for service robots. Recently object recognition has been dominated by the use of Deep Neural Networks (DNN), however, they required a large number of images and long training times. If a user asks a service robot to search for an unknown object, it has to deal with selecting relevant images to learn a model, deal with polysemy, and learn a model relatively quickly to be of any use to the user. In this paper we describe an object recognition system that deals with the above challenges by: (i) a user interface to reduce different object interpretations, (ii) downloading on-the-fly images from Internet to train a model, and (iii) using the outputs of a trimmed pre-trained DNN as attributes for a SVM. The whole process (selecting and downloading images and training a model) of learning a model for an unknown object takes around two minutes. The proposed method was tested on 72 common objects found in a house environment with very high precision and recall rates (over 90%).

1 Introduction

Object recognition is an open research area based on obtaining models able to identify one kind of object from another. This task has several applications, for example, for service robotics, which must be capable of finding, ideally, any object requested by a user. However, accomplishing this assignment involves many problems such as the great variety of existing objects of the same kind, *e.g.*, four chairs are depicted in Fig. 1, every one of them has unique characteristics that make it different from the others, but at the end, all of them are chairs.

On the other hand, also the polysemy problem must be considered. This problem happens when a word has more than one meaning, *e.g.*, the word "mouse" can refer both, the animal or the computer device. Then, for these cases a robot can not only learn the general concept, it must learn the precise meaning of the concept according to what the user requested.

Therefore, given all the possible requests that a user can ask and all the meanings for each request, it is intractable to train a model for this unthinkable amount of concepts. Instead, models able to differentiating the correct meaning

F. Castro et al. (Eds.): MICAI 2017, LNAI 10632, pp. 41–53, 2018.
https://doi.org/10.1007/978-3-030-02837-4_4

Fig. 1. Chairs with different characteristics. Images retrieved from: https://www.pinterest.com.mx

of the user's request and learning quickly any object's characteristics in order to recognize them are necessary.

Nowadays, the most accurate methods to recognize objects are those using Deep Neural Networks (DNN). Since Krizhevsky *et al.* (2012) [1] won the ILSVRC-2012 with their DNN called AlexNet, many modifications to this network and also new architectures have been proposed in order to improve the object recognition performance [2–8], however, training these networks takes months. Imagine an scenario where the user requests a service robot to find an unknown object. In this case, the robot must learn a model of that object as quickly as possible in order to recognize it and fulfill the user's request.

This research proposes an object recognition system suitable for service robots and focused on both, polysemy and fast learning. The first is managed through an interface where the user can refine his/her request until it is clear the type of object that must be learned. The latter is dealt through transfer learning and downloading the training images directly from Internet. The output of one of the DNN's last layers is used as the attributes to train a Support Vector Machine (SVM) with images that are downloaded on-the-fly to recognize new concepts.

Summarizing, the proposed system allows users to specify what they want and uses transfer learning for quickly learning accurate models for recognizing previously unknown objects.

The rest of the article is organized as follows. In Sect. 2 related work is revised. In Sect. 3 our object recognition system is presented in detail. In Sect. 4 the experiments and the results obtained are described. Finally, in Sect. 5 the conclusions and the future work of this research are discussed.

2 Related Work

DNNs have been used in the last years for solving different recognition tasks. The most common DNNs are the AlexNet, CaffeNet [3], VGG (with its variants) [4,5] and Inception v3 [8]. The four DNNs have been pre-trained using ImageNet, and these models and the networks ready to be trained are available for implementing.

Training a DNN from scratch can take several weeks in the best cases. Because of that, previous works have tried to speed-up the DNN's training time

using different techniques, such as: optimization [9–11], Fourier domain [12], networks compressing [13,14] and fine-tuning [15,16], among others.

For the latter mentioned methods, fine-tuning is the faster way to train a network accomplishing the task in less than an hour. This approach is based on the idea of retrained the last layers of the network maintaining the rest without changes. The use of outputs of intermediate layers as attributes has been previously studied. This method can be considered as transfer learning [17] due to it uses the features transferred from the unaltered part of the network as the input for layers to be retrained.

Fine-tuning have been successfully used classifying images from completely different domains such as land segmentation [16,18], emotion recognition [19] or medical images [20]. However, the retraining time required by this method remains useless if we need an on-line training system that can be used by service robots. In order to decrease the time to retrain a network, the proposed system is based on the transfer learning idea, obtaining a layer output as the attributes of a SVM which can be quickly retrained.

Some works, based on the same principle as our system, are shown in Table 1. They use a DNN's layer output as the SVM's attributes, however, they present some differences with our system. The characteristics analyzed for each work listed in Table 1 are: the DNN used; if the training happens off-line or on-line; the classifier receiving the transfered features; and, the application.

Table 1. Related works using transfer learning

	DNN	Training	Transfer learning	Application
Ng *et al.* (2015) [19]	AlexNet and VGG-CNN-M	Off-line	Double fine-tuning	Emotions recognition
Codella *et al.* (2015) [21]	CaffeNet	Off-line	Sparse Coding + SVM	Melanoma recognition
Akilan *et al.* (2017) [22]	AlexNet, VGG-16 and Inception v3 (all together)	Off-line	SVM	Object and action recognition
D'Innocente *et al.* (2017) [23]	Re-trained AlexNet and Inception v3	Off-line	SVM	Objects recognition
Our system	Inception v3	On-line	SVM	Home objects recognition

As can be seen, the applications for all the works presented in Table 1 are recognition tasks, however all of them, except us, are re-trained off-line. Codella *et al.* (2015) [21] added the features transfered from the DNN to the features obtained from a Sparse Coding for the SVM's input features, while Akilan *et al.* (2017) [22] used a system combining the outputs of three DNNs as SVM's attributes. These methods need much more processing time.

On the other hand, D'Innocente *et al.* (2017) [23] presented the most closely related work to our system, however, they need to retrain the complete DNN to

accomplish the object recognition task. Also, the images used for training our system are directly downloaded on-the-fly from Internet.

Summarizing, the main advantage of our system against related work is its capacity to be retrained on-line with images selected on-the-fly without losing accuracy in the recognition of new objects.

3 Object Recognition System

The proposed object recognition system is composed by six steps as shown in Fig. 2. At the beginning, the system receives as input the concept of the object to be recognized and then the process starts.

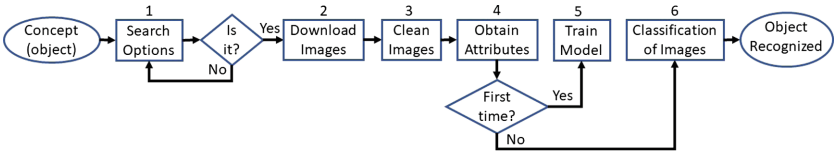


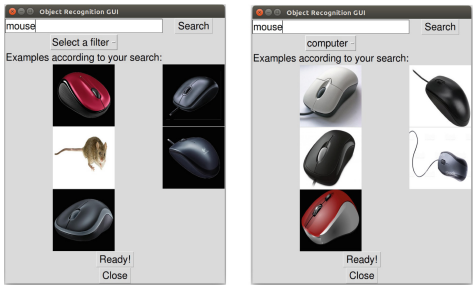
Fig. 2. Flowchart of the object recognition system

Step 1 searches images, of the concept requested by the user, to train the system using *Google Images*. In order to deal with polysemy, an interface, depicted in Fig. 3, is shown to the user with possible options related with the concept in order to filter the search if needed. The user must select the most related option with his/her request or introduce a more specific concept if all the options are unsuitable. When the new concept or a filter is selected the interface shows the new possible options considering the refined search.

Figure 3(a) shows an example when the requested concept is "mouse". As can be seen, the example images consider both, animals and computer devices. However, if the filter "computer" is selected, the search is refined and now all the example images correspond to computer devices as shown in Fig. 3(b)

Step 2 uses the concept selected in the previous step as the search string to download 100 images from *Google Images* to train the system. As the previous step considers the concept's polysemy, the downloaded images are expected to be more related with the concept that must be learned.

Step 3 consists of "cleaning" the downloaded images, i.e., making the format and size of the images suitable to be used in next steps. For Steps 4, 5 and 6, three methods were explored: the pre-trained *Iv3* deep neural network; a SVM classifier using images' color and shape features as attributes; and a SVM classifier using an output layer from *Iv3* as attributes. These methods are explained in detail below.



(a) (b)

Fig. 3. Interface used to show users the example images according to his/her request: (a) when the concept "mouse" is requested; (b) when the filter "computer" is added to the original request

3.1 Pre-trained Iv3 Deep Neural Network (*Iv3* Method)

The Iv3 model is a deep neural network with 42 layers, a representation of this network is depicted in Fig. 4. This model is available in the open source software library *TensorFlow* (TF), where it has been trained for the ImageNet Large Visual Recognition Challenge using the data from 2012. For this task, the Iv3 should classify images into 1000 classes considering daily life objects, animals, food, or means of transport, among others.

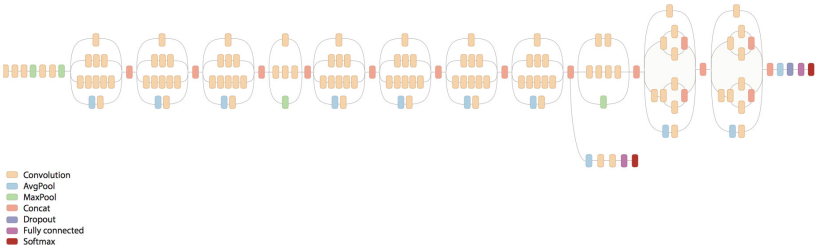


Fig. 4. Inception v3 general architecture

For the proposed object recognition system, the pre-trained Iv3 model has been selected over other commonly used DNN models such as AlexNet, CaffeNet, and VGG. Iv3 requires less computational resources [7], which makes it feasible for being used in on-line applications, as the implemented in this work. Furthermore, it outperforms the object recognition precision achieved by the other DNN models [22].

The Iv3 model is obtained from TF and it deals with the Steps 4, 5 and 6 of the proposed object recognition system. First, for Step 4, it obtains the attributes for the images from Step 3. For this method, the images need both,

being in *jpg* format and having a size of 299×299 pixels. Then, since the model is pre-trained, Step 5 is not needed. Finally, the last 3 layers of the model perform the classification using the attributes obtained from Step 4 in the dropout and fully connected layers, and in the *Softmax* classifier for 1,000 classes. Therefore, using this method, it is possible to recognize the objects belonged to the classes for which the model was trained.

3.2 SVM Classifier Using Color and Shape Features ($Color + Shape + SVM$ Method)

This method implements the last three steps of the proposed system. For Step 4, histograms of oriented gradients [24] and dominant color identification features of objects are extracted.

Using these features as attributes, a SVM is trained in Step 5. Finally, the SVM is used to predict classes in Step 6 and, therefore, to recognize objects obtaining the shape and color features of the images tested. This method can be applied to any requested object.

3.3 SVM Classifier Using Iv3 ($Iv3 + SVM : All$ Method)

This method uses the Iv3 model to obtain the images' attributes for Step 4 of the object recognition system. As it was mentioned above, Iv3 has 42 layers and the last 3 (dropout, fully connected and *Softmax* layers) are used to perform the classification. It has been previously analysed how the features used to describe images on a DNN goes from the most general to the most specific as the DNN's output is closer [17]. Moreover, it has been proved that better results are obtained transferring features from the DNN's layers before the fully connected layers [23].

In this regard, the input vector for the Iv3 model's dropout layer, which has a size of 2,048, is considered as the set of attributes which describes an image independently of its domain. Therefore, for this method, Step 4 trims the Iv3 network just after the last average pool layer and interprets the resulting vector as the image descriptors that will be used as the attributes for training a SVM classifier in Step 5, as shown in Fig. 5.

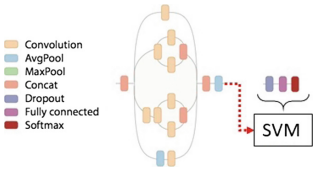


Fig. 5. SVM replacing the last three layers of the Inception v3 architecture

After training, images can be processed using the Iv3 model to obtain their attributes and the trained SVM will be used as Step 6 of the object recognition

system. For this method, since the attributes are obtained from the Iv3 model, the images also need both, being in *jpg* format and having a size of 299×299 pixels. Applying this method it is also possible to recognize any object requested by the user.

Binary Classification (*Iv3 + SVM : Binary* Method). For this method it is also proposed a variant where the model is trained for only two classes, a positive and a negative, where the positive class examples are those representing the searched object and the negative class examples are those representing any other object different to the positive class.

Therefore, for this variant, Step 4 remains the same, Step 5 will always be performed for two classes and Step 6 will say whether or not the object is the one it has been searching for.

4 Experiments and Results

The performance of the proposed object recognition system depends on the accuracy of the classification. Therefore, the first experiment conducted was a classification task comparing the precision, recall and f-measure obtained by the three methods proposed against the $Iv3$ method when classifying 72 classes. These objects were selected by 7 independent users that were asked to suggest 10 objects that can be found by a service robot in a house environment, and complemented with objects from the Semantic Robot Vision Challenge 2009 and ImageCLEF2014.

A dataset of 5,670 images obtained from *Google Images* (80 images per class) was created for this task. Implementation of the Iv3 model was accomplished through *TensorFlow*. On the other hand, the SVM was implemented using Sklearn from the Scikit package for Python. The classification using the SVM was performed applying a 5 fold-cross validation (80% examples for training, 20% examples for testing).

The classes involved in the experiment are restrained to objects that can be found in a house. Some of these 72 classes are among the 1,000 classes used to train the Iv3 model, but not all of them. In Table 2 are shown all the classes involved in the experiment. These classes are divided into three groups according with their correspondence with the classes considered by the Iv3 model.

In the first column are the classes with a Direct Correspondence (DC) between our classes and the Iv3 classes, *i.e.*, those classes that are the same in both groups.

In the second column are the classes with an Indirect Correspondence (IC) between both groups, *i.e.*, those of the 72 classes representing a more general class than others considered by the Iv3 model, *e.g.*, the classes "tennis", "sandals", "boots", or "heels", considered by the Iv3 model, are represented with our class "Shoe", therefore, if the Iv3 model classifies an example as any kind of "Shoe", it will be considered as correct because all are shoes.

Finally, in the third column are the classes with No Correspondence between both groups (NC). Therefore, these classes are not considered in the results reported for the $Iv3$ method.

Table 2. Classes with a direct correspondence (DC), an indirect correspondence (IC) and without correspondence (NC)

DC classes	IC classes		NC classes	
Pen	Apple	Shirt	Book	Blanket
Cellphone	Shoe	Sock	Spoon	Milk
Towel	Coffee	Cup	Fork	Soap
Tv remote	Clothe	Knife	Glass of water	Pumpkin
Broom	Laptop	Coat	Coke	Dinosaur toy
Mop	Bread	Bottle	Key	CD
Backpack	Bed sheet	Toy car	Scissor	DVD
Pillow	Handbag	Potato chips	Comb	Video game
Glasses	Food	Phone	Bible	Toy
Orange	Chair	Bookshelf	Cell charger	Cracker
Paddle	Medicine		Baby bottle	Cookie
Soccer ball	Ac remote		Headphone	Extinguisher
Frying pan	Tablet		Nail clipper	Potty
Printer	Plate		Inhaler	Softener
Trash	Jacket		Cosmetic bag	
Fridge	Hand cream		Fly swatter	

The average results for precision, recall and f-measure obtained by each method are shown in Table 3. From the second to the fifth columns the results for the $Iv3$, $Color + Shape + SVM$, $Iv3 + SVM : All$ and $Iv3 + SVM : Binary$ methods are presented, respectively.

As can be seen, the best results are obtained by the $Iv3 + SVM : Binary$ method. This method has a performance of at least 90% for the three measurements and for all the groups of classes. The latter lead us to make some remarks.

First of all, the performance of the $Iv3 + SVM : Binary$ method is slightly better for the DC and IC classes than the NC classes, however, this difference is lower than 3% for precision, recall and f-measure. This proves that the method is consistent whether the recognized objects have or not a correspondence with the classes for which the Iv3 model was trained. This is important because it means that we can quickly learn highly accurate models for objects found in house environments using features of a DNN that was not necessarily trained on those objects.

Also, it should be noted that there is a significant increase in performance when using a binary classification. The classification precision for the $Iv3$ method was improved in almost 10% for the AC classes (DC + IC classes), whereas the recall was improved in more than 20%. Although the $Iv3$ was trained for 1,000 classes and we are comparing with classifiers of 72 classes, we expect the robot to be using mainly binary classifiers.

Table 3. Average precision, recall and f-measure for the four methods

Group of classes	Measurement	*Iv3*	*Color + Shape + SVM*	*Iv3 + SVM : All*	*Iv3 + SVM : Binary*
AC (DC+IC)	Precision	0.894	0.085	0.884	0.986
	Recall	0.703	0.122	0.851	0.932
	F-Measure	0.770	0.075	0.865	0.958
DC	Precision	0.926	0.060	0.913	0.987
	Recall	0.755	0.160	0.873	0.934
	F-Measure	0.822	0.070	0.892	0.959
IC	Precision	0.874	0.100	0.866	0.985
	Recall	0.671	0.100	0.838	0.931
	F-Measure	0.738	0.079	0.849	0.957
NC	Precision	—	0.091	0.802	0.967
	Recall	—	0.087	0.819	0.898
	F-Measure	—	0.077	0.807	0.930

On the other hand, the $Iv3 + SVM : All$ method have a 1% lower precision than the $Iv3$, which means that substituting the last layers of the pre-trained DNN by a SVM does not decrease considerably the classification precision. Furthermore, the f-measure increase almost 10% using the $Iv3+SVM : All$ method, due to the better recall results. Furthermore, the $Iv3+SVM : All$ has the advantage to be easily re-trained for recognizing any object among a group.

The performance for both methods, $Iv3$ and $Iv3+SVM : All$, is quite similar, however, statistical significance of differences was computed and proved using a χ^2 test with a 2×2 contingency table and $\alpha = 0.05$, obtaining a $p - value < 0.0001$ for the AC, DC and IC groups of classes. This test assumes unpaired data which corresponds to our situation since the training sets were different for each method.

Finally, it was surprising the $Color + Shape + SVM$ method's poor performance. During some previous experiments with cleaner images and fewer classes, this method showed a good performance. However, our dataset from *Google Images* contains images that are not exactly related with the classes or

that have many different views, and these could be some of the factors causing the drop of the method's performance.

Given the outcomes of the first experiment, the $Iv3+SVM : Binary$ method was selected to complete our object recognition system. The Iv3 model is the top of the object recognition solutions until now, and the main goal of our experiment was not to propose a better object recognition system, but instead, to demonstrate that applying our method it is feasible to recognize, with a high precision, an unknown object using transfer learning from a pre-trained DNN.

On the other hand, the second experiment conducted consists of measuring the execution time elapsed by the entire object recognition system using our selected method. The results of this experiment are shown in Table 4. The second column of the table depicts the number of images involved in every step of the system and the third column shows the average time elapsed for each step after 10 complete runs.

The object recognition system runs on Python 2.7.12 over Ubuntu 16.04. The computer used for the experiments has four Intel Core i7-6500U processors at 2.50 GHz and 8 Gb RAM. The Internet connection speed for downloads was 10 Mb/s.

Table 4. Execution time for the entire object recognition system using the $Iv3+SVM : Binary$ method

	Images involved	Iv3 + SVM : Binary
Step 1: Search options	6	14.04 ± 2.81 s
Step 2: Download images	80	77.23 ± 3.29 s
Step 3: Clean images	$80 \rightarrow 75$	2.16 ± 0.06 s
Step 4: Obtain attributes	150	33.40 ± 1.00 s
Step 5: Train the model	150	0.12 ± 0.02 s
Step 6: Classify images	1	1.10 ± 0.32 s
Total time		128.06 ± 3.50 s

The first step involves 6 example images used to show what the user can expect according to his/her search string. The Step 1 duration will vary depending on the Internet connection speed, but in our case, it will remain less than 20 s whichever the object to be searched or the filters applied.

The second step involves the images used to learn a concept, for our experiments, we used 80 images. Again, the duration of this step depends on the Internet connection speed, in our case, the images were approximately downloaded at a rate of 1 image per second. The images' size and the number of images used for the model training impact on the time required to finish this step. However, for our experiments, 80 images were enough to obtain high classification rates as it was shown in the first experiment.

The third step consists of cleaning the 80 downloaded images, since *TensorFlow* works only with *jpg* format images. Images from *Google Images* have different formats and not all of them can be retrieved correctly. For each one of the 10 runs of our experiment, the cleaning step removed 5 images that were unusable. Therefore, at the end of the step, there are 75 images ready to be processed in order to obtain their attributes. The time consumed by this step was very constant along all runs, and it was always less than 2.3 s.

The $Iv3 + SVM : Binary$ method starts working on Step 4 where attributes from the 75 downloaded and cleaned images must be obtained. These examples are labeled as the positive class. On the other hand, 75 negative class examples are selected from images of the 72 classes involved in our first experiment. If the positive class is among our 72 classes, its examples are discarded from the dataset. The time rate to retrieve the images' attributes using the Iv3 DNN was on average 2.25 images per second. Therefore, using this method, it is possible to create, in less than 35 s, a data set able to train an accurate object recognition system.

Step 5 is the faster on the entire system as the SVM requires less than 0.15 s to be trained. Therefore, aggregating Steps 4 and 5, our system is able to learn a new concept in less than 36 s, and then, recognize what was learned with at least 96% of precision.

Finally, once the concept was learned, the system is ready to search the required object. Thus, our system will take less than 1.5 s evaluating an object in order to determine if it is or it is not the requested object.

Overall, the proposed system is able to learn an accurate model of a completely unknown object using images from Internet in roughly two minutes.

5 Conclusions and Future Work

Object recognition is a relevant task in computer vision and in particular for robotics. The field has been recently dominated by Deep Neural Networks achieving impressive results over a large number of objects. Training DNN models, however, require a large number of samples and take very long training times. If we want a robot to search for an unknown object it has to select relevant images and quickly induce an adequate model to be useful for the user. In this paper we described a system that downloads relevant images from Internet, with the aid of a user interface, and leverages the use of DNNs to quickly learn a model using the outputs of pre-trained intermediate layers as features of a fast classifier. It is shown that the proposed system can learn a model for an unknown object in about two minutes with high performance measures, even in objects that were not used for training the original DNN.

As future work, we will like to use a real robot to search for unknown objects, suggested by independent users, in a house environment.

References

1. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
2. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun, Y.: OverFeat: integrated recognition, localization and detection using convolutional networks. arXiv preprint arXiv:1312.6229 (2013)
3. Jia, Y., et al.: Caffe: convolutional architecture for fast feature embedding. In: Proceedings of the 22nd ACM International Conference on Multimedia, pp. 675–678. ACM (2014)
4. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
5. Chatfield, K., Simonyan, K., Vedaldi, A., Zisserman, A.: Return of the devil in the details: delving deep into convolutional nets. arXiv preprint arXiv:1405.3531 (2014)
6. Ouyang, W., et al.: DeepID-Net: deformable deep convolutional neural networks for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2403–2412 (2015)
7. Szegedy, C., et al.: Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–9 (2015)
8. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826 (2016)
9. Jaderberg, M., Vedaldi, A., Zisserman, A.: Speeding up convolutional neural networks with low rank expansions. arXiv preprint arXiv:1405.3866 (2014)
10. Lebedev, V., Ganin, Y., Rakhuba, M., Oseledets, I., Lempitsky, V.: Speeding-up convolutional neural networks using fine-tuned CP-decomposition. arXiv preprint arXiv:1412.6553 (2014)
11. Zhang, X., Zou, J., He, K., Sun, J.: Accelerating very deep convolutionalnetworks for classification and detection. IEEE Trans. Pattern Anal. Mach. Intell. **38**(10), 1943–1955 (2016)
12. Mathieu, M., Henaff, M., LeCun, Y.: Fast training of convolutional networks through FFTs. arXiv preprint arXiv:1312.5851 (2013)
13. Kim, Y.-D., Park, E., Yoo, S., Choi, T., Yang, L., Shin, D.: Compression of deep convolutional neural networks for fast and low power mobile applications. arXiv preprint arXiv:1511.06530 (2015)
14. Chen, W., Wilson, J., Tyree, S., Weinberger, K., Chen, Y.: Compressing neural networks with the hashing trick. In: International Conference on Machine Learning, pp. 2285–2294 (2015)
15. Carneiro, G., Nascimento, J., Bradley, A.P.: Unregistered multiview mammogram analysis with pre-trained deep learning models. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 652–660. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_78
16. Castelluccio, M., Poggi, G., Sansone, C., Verdoliva, L.: Land use classification in remote sensing images by convolutional neural networks. arXiv preprint arXiv:1508.00092 (2015)
17. Yosinski, J., Clune, J., Bengio, Y., Lipson, H.: How transferable are features in deep neural networks? In: Advances in Neural Information Processing Systems, pp. 3320–3328 (2014)

18. Penatti, O.A., Nogueira, K., dos Santos, J.A.: Do deep features generalize from everyday objects to remote sensing and aerial scenes domains? In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 44–51 (2015)
19. Ng, H.-W., Nguyen, V.D., Vonikakis, V., Winkler, S.: Deep learning for emotion recognition on small datasets using transfer learning. In: Proceedings of the ACM International Conference on Multimodal Interaction, pp. 443–449. ACM (2015)
20. Litjens, G., et al.: A survey on deep learning in medical image analysis. arXiv preprint arXiv:1702.05747 (2017)
21. Codella, N., Cai, J., Abedini, M., Garnavi, R., Halpern, A., Smith, J.R.: Deep learning, sparse coding, and SVM for melanoma recognition in dermoscopy images. In: Zhou, L., Wang, L., Wang, Q., Shi, Y. (eds.) MLMI 2015. LNCS, vol. 9352, pp. 118–126. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24888-2_15
22. Akilan, T., Wu, Q.J., Yang, Y., Safaei, A.: Fusion of transfer learning features and its application in image classification. In: IEEE 30th Canadian Conference on Electrical and Computer Engineering, pp. 1–5. IEEE (2017)
23. D'Innocente, A., Carlucci, F.M., Colosi, M., Caputo, B.: Bridging between computer and robot vision through data augmentation: a case study on object recognition. arXiv preprint arXiv:1705.02139 (2017)
24. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 886–893. IEEE (2005)