



DeepCompareJ: Comparing Image Classification Models

A. Inés^(✉), C. Domínguez, J. Heras, E. Mata, and V. Pascual

Department of Mathematics and Computer Science,
University of La Rioja, Logroño, Spain
{adrian.ines,cesar.dominguez,jonathan.heras,eloy.mata,
vico.pascual}@unirioja.es

Abstract. Image classification is a computer vision task that has several applications in diverse fields like security, biology or medicine; and, currently, deep learning techniques have become the state-of-the-art to create image classification models. This growing use of deep learning techniques is due to the large amount of data, the fast increase of the computer processing capacity, and the openness of deep learning tools. However, whenever deep learning techniques are used to solve a classification problem, we can find several deep learning frameworks with their own peculiarities, and different models in each framework; hence, it is natural to wonder which option fits better our problem. In this paper, we present DeepCompareJ, an open-source tool that has been designed to compare, with respect to a given dataset, the quality of deep models created using different frameworks.

Keywords: Deep learning · Image classification · Integration

1 Introduction

Image classification is an instrumental tool to solve several problems in diverse fields like security, biology or medicine; and, deep learning techniques have become the state-of-the-art approach to deal with them. Just to name a few examples, deep learning techniques have been applied for classifying objects within X-ray baggage [1], for detecting parasite malaria in thin blood smear images [16], for classifying breast cancer histology images [2], or for classifying skin cancer images [8].

The reason for the success of deep learning techniques is threefold: the fast increase of the computer processing capacity, the large amount of data, and the openness of deep learning methods. Namely, deep learning frameworks—such as Keras [4], Caffe [11] or MxNet [3]—are open-source libraries, and most of the models built with them are freely distributed [14]. Due to its open nature,

Supported by Ministerio de Industria, Economía y Competitividad, project MTM2017-88804-P; Agencia de Desarrollo Económico de La Rioja, project 2017-I-IDD-00018; and FPU Grant 16/06903 of the Spanish MEC.

© Springer Nature Switzerland AG 2020
R. Moreno-Díaz et al. (Eds.): EUROCAST 2019, LNCS 12014, pp. 256–262, 2020.
https://doi.org/10.1007/978-3-030-45096-0_32

jonathan.heras@unirioja.es

whenever we want to use a deep learning model to solve an image classification problem, we can find several models that can be adapted to solve the same task; hence, it is natural to wonder which model better fits to our dataset; or whether a new model that we have trained is more accurate than the existing ones. However, each deep learning framework has its own peculiarities, for instance, most frameworks can be used from different programming languages like Python or C++. However, as we can expect, a common language for all the deep learning frameworks does not exist. Hence, we have to pay attention to which programming languages support the framework we want to use. Also, the representation of the images depends on the framework, some frameworks represent the images with an image class like PIL, others for example represent the images with arrays or tensors. In addition, the representation of the layers differs from one framework to another, or even not all the frameworks have implemented all the kinds of layers. Because of this, some models can not be implemented in all the frameworks. These are some examples of the differences we can find among different deep learning frameworks. Therefore, it is difficult to work with several deep learning frameworks at the same time or even compare the quality of several models from different frameworks.

In this work, we have tackled this problem by developing DeepCompareJ, an open-source tool, built on top of DeepClas4Bio, that allows users to easily compare models from different deep learning frameworks without worrying about the particularities of each library.

2 DeepClas4Bio and DeepCompareJ

DeepClas4Bio [10] is an extensible API that provides a common access point for classification models of several deep learning frameworks. This API allows users to work with those deep learning frameworks and models easily and transparently for them. Namely, the DeepClas4Bio API provides five public methods, as shown in Fig. 1.

The method called `listFrameworks` allows users to know the frameworks included in the API. Currently, the API gives support for Keras [4], Caffe [11], DeepLearning4J [7], PyTorch [15] and MxNet [3]. In addition, for each framework in the API, we can find several models using the method `listModels`, that provides the available deep models for a particular framework; for instance, the networks VGG [17], AlexNet [13], ResNet [9] and GoogleNet [18], trained for the ImageNet challenge [6], are available in Keras. Also, the API allows users to classify an image or a batch of images given a specific model and framework using respectively the `predict` or the `predictBatch` method. Finally, we can find a method called `evaluate`, that can be used to compare the quality of several models with respect to a dataset using different measures (accuracy, rank5, precision, recall, F1-score, Jaccard Index, Matthews Correlation and AUROC are available).

DeepClas4Bio API has several applications: classifying images or a batch of images (several examples can be seen in the project webpage), evaluating the

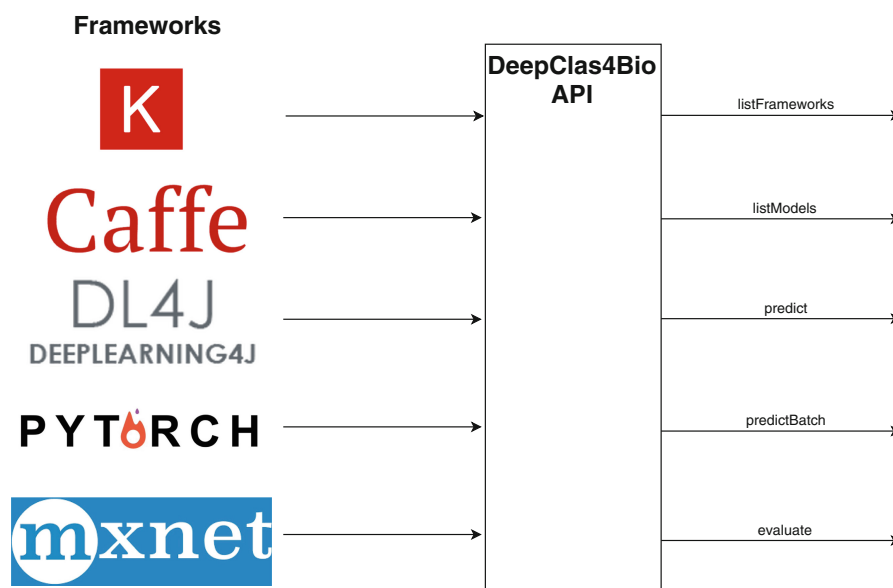


Fig. 1. Structure of the DeepClas4Bio project.

quality of a model; or comparing the quality of several models. In this paper, we focus on the latter application. In particular, we have developed DeepCompareJ, an open-source Java application built on top of DeepClas4Bio.

In order to compare image classification models using DeepCompareJ, the user must select: (1) the models to compare; (2) the measures to evaluate the models; and, (3) the dataset of images. DeepCompareJ offers several models from different deep learning frameworks, several measures and several procedures to load an annotated dataset of images (by default, the API loads datasets from a folder where each class has its corresponding folder of images). All this information can be selected from the interface of Fig. 2, and it is obtained from the DeepClas4Bio API.

The workflow of DeepCompareJ is straightforward. When DeepCompareJ starts, the application connects with the DeepClas4Bio API to obtain all the models, measures and ways of loading datasets available in the system. Then, the users select the most suitable options for their problems; and, finally, DeepCompareJ connects with the DeepClas4Bio API to compare the selected models and shows the result in a table.

3 Case Studies

In this section, we show a couple of examples of the usage of DeepCompareJ. In the first example, we compare several models trained in the Imagenet challenge [6]. And, in the second example, we compare models trained in the Kvasir dataset [12], a gastrointestinal disease dataset.

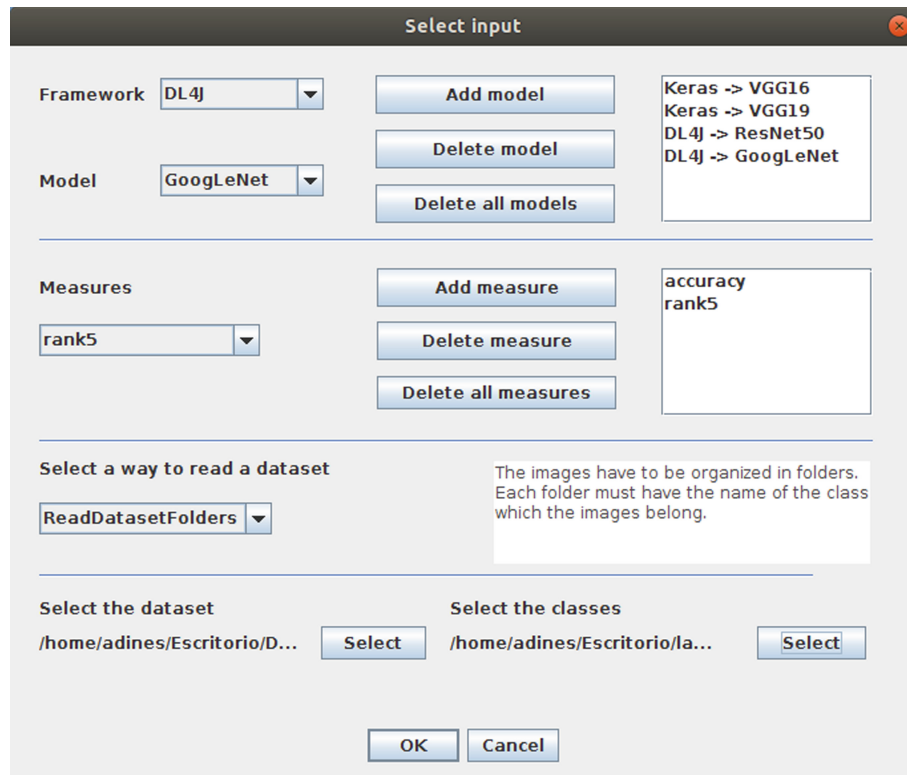
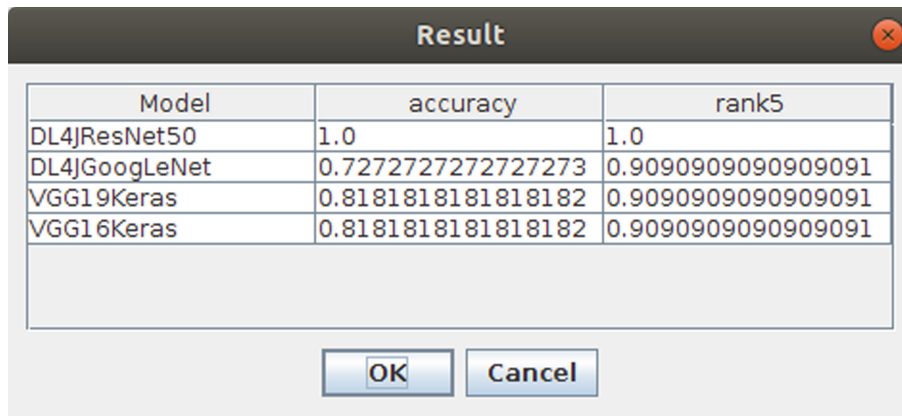


Fig. 2. Interface of DeepCompareJ.

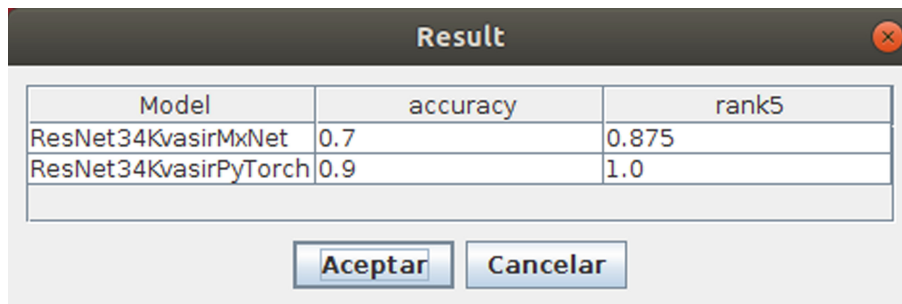
Namely, in the first example we have compared the quality of four models trained in the Imagenet dataset. The four models used in this example are ResNet50 and GoogleNet from DeepLearning4J—a Java framework for deep learning—and VGG16 and VGG19 from Keras—an open source neural network library written in Python, capable of running on top of TensorFlow, CNTK, or Theano. The measures selected to compare these models are accuracy and rank5. Finally, the dataset used to compare these models is a subset of the Imagenet dataset. The result of this experiment is shown in Fig. 3.

In the second example, we have compared the quality of two models trained in the Kvasir dataset. The models used in this example are ResNet34 trained in the MxNet framework—a deep learning framework that gives support for several programming languages, namely Python, Scala, R, Julia and Perl—and also ResNet34 trained in the PyTorch framework—a Python deep learning framework based on the Torch library [5]. The measures selected to compare these models are accuracy and rank5. The results of this experiment are shown in Fig. 4.



Model	accuracy	rank5
DL4JResNet50	1.0	1.0
DL4JGoogLeNet	0.72727272727273	0.90909090909091
VGG19Keras	0.81818181818182	0.90909090909091
VGG16Keras	0.81818181818182	0.90909090909091

Fig. 3. Results table produced by DeepCompareJ for a subset of the Imagenet challenge.



Model	accuracy	rank5
ResNet34KvasirMxNet	0.7	0.875
ResNet34KvasirPyTorch	0.9	1.0

Fig. 4. Results table produced by DeepCompareJ for a subset of the Kvasir dataset.

4 Conclusions

In this paper, we have presented DeepCompareJ, an open source tool that can be used to easily compare the quality of several deep classification models. This tool solves the problem of comparing models from different deep learning frameworks using the same dataset thanks to the DeepClas4Bio API.

Another problem solved by DeepCompareJ and DeepClas4Bio is the task of working with different frameworks at the same time. This is due to the fact that deep learning frameworks can have several differences, for example, the programming language, the supported layers or the way to load an image. DeepClas4Bio and DeepCompareJ abstract these particularities and hide these difficulties to the user improving the usability of different deep learning frameworks.

5 Availability and Requirements

- Project name: DeepCompareJ.
- Project home page: <https://github.com/adines/DeepCompareJ>.
- Operating system(s): Platform independent.
- Programming language: Java.
- License: GNU GPL 3.0.
- Any restrictions to use by non-academics: None.
- Dependencies: DeepClas4Bio (<https://github.com/adines/DeepClas4Bio>).

The project home page contains the installation instructions and usage examples.

References

1. Akay, S., Kundegorski, M.E., Devereux, M., Breckon, T.P.: Transfer learning using convolutional neural networks for object classification within X-ray baggage security imagery. In: 2016 IEEE International Conference on Image Processing (ICIP), pp. 1057–1061 (2016)
2. Araújo, T., Aresta, G., Castro, E., Rouco, J., Aguiar, P., Eloy, C., et al.: Classification of breast cancer histology images using convolutional neural networks. PLoS ONE **12**(6), e0177544 (2017)
3. Chen, T., et al.: MxNet: a flexible and efficient machine learning library for heterogeneous distributed systems. In: Proceedings of Neural Information Processing Systems (NIPS 2015) - Workshop on Machine Learning Systems (2015)
4. Chollet, F.: Keras: the Python deep learning library (2015). <https://keras.io>
5. Collobert, R., Bengio, S., Mariéthoz, J.: Torch: a modular machine learning software library. Technical report, IDIAP (2002)
6. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: a large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255 (2009)
7. Eclipse Deeplearning4j Development Team: Deeplearning4j: Open-source, distributed deep learning for the JVM (2018). <https://deeplearning4j.org/>
8. Esteva, A., et al.: Dermatologist-level classification of skin cancer with deep neural networks. Nature **542**, 115–118 (2017)
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016), pp. 770–778 (2016)
10. Inés, A., Domínguez, C., Heras, J., Mata, E., Pascual, V.: DeepClas4Bio: connecting bioimaging tools with deep learning frameworks for image classification. Comput. Biol. Med. **108**, 49–56 (2019)
11. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: convolutional architecture for fast feature embedding. In: Proceedings of the 22nd ACM International Conference on Multimedia. ACM (2014)
12. Pogorelov, K., Randel, K.R., Griwodz, C., Eskeland, S.L., de Lange, T., Johansen, D., et al.: KVASIR: a multi-class image dataset for computer aided gastrointestinal disease detection. In: Proceedings of the 8th ACM on Multimedia Systems Conference, MMSys 2017, pp. 164–169. ACM, New York (2017). <https://doi.org/10.1145/3083187.3083212>

13. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q. (eds.) *Advances in Neural Information Processing Systems 25*, pp. 1097–1105. Curran Associates, Inc. (2012)
14. ModelZoo: Model Zoo: Discover open source deep learning code and pretrained models (2018). <https://modelzoo.co/>
15. Paszke, A., et al.: Automatic differentiation in PyTorch. In: *Proceedings of Neural Information Processing Systems (NIPS 2017) - Workshop* (2017)
16. Rajaraman, S., et al.: Pre-trained convolutional neural networks as feature extractors toward improved malaria parasite detection in thin blood smear images. *PeerJ* (2018). <https://lhncbc.nlm.nih.gov/system/files/pub9752.pdf>
17. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: *Proceedings of International Conference on Learning Representation (ICLR 2015)* (2015)
18. Szegedy, C., et al.: Going deeper with convolutions. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2015)*, pp. 1–9 (2015)