Reproducibility of Deep CNN for Biomedical Image Processing Across Frameworks and Architectures

Stefano Marrone
University of Naples Federico II
Naples, Italy
stefano.marrone@unina.it

Stefano Olivieri
The MathWorks Srl
Tourin, Italy
stefano.olivieri@mathworks.it

Gabriele Piantadosi
University of Naples Federico II
Naples, Italy
gabriele.piantadosi@unina.it

Carlo Sansone
University of Naples Federico II
Naples, Italy
carlo.sansone@unina.it

***Abstract*—With the increasing spread of easy and effective frameworks, in recent years Deep Learning approaches are becoming more and more used in several application fields, including computer vision (such as natural and biomedical image processing), automatic speech recognition (ASR) and time-series analysis. If, on one hand, the availability of such frameworks allows developers to use the one they feel more comfortable with, on the other, it raises questions related to the reproducibility of the designed model across different hardware and software configurations, both at training and at inference times. The reproducibility assessment is important to determine if the resulting model produces good or bad outcomes just because of luckier or blunter environmental training conditions. This is a non-trivial problem for Deep Learning based applications, not only because their training and optimization phases strongly rely on stochastic procedures, but also because of the use of some heuristic considerations (mainly speculative procedures) at training time that, although they help in reducing the required computational effort, tend to introduce non-deterministic behavior, with a direct impact on the results and on the model's reproducibility. Usually, to face this problem, designers make use of probabilistic considerations about the distribution of data or focus their attention on very huge datasets. However, this kind of approach does not really fit some application field standards (such as medical imaging analysis with Computer-Aided Detection and Diagnosis systems – CAD) that require strong demonstrable proofs of effectiveness and repeatability of results across the population. It is our opinion that in those cases it is of crucial importance to clarify if and to what extent a Deep Learning based application is stable and repeatable as well as effective, across different environmental (hardware and software) configurations. Therefore, the aim of this work is to quantitatively analyze the reproducibility problem of Convolutional Neural Networks (CNN) based approaches for the biomedical image processing, in order to highlight the impact that a given software framework and hardware configurations might have when facing the same problem by the same means. In particular, we analyzed the problem of breast tissue segmentation in DCE-MRI by using a modified version of a 2D U-Net CNN, a very effective deep architecture for semantic segmentation, using two Deep Learning frameworks (MATLAB and TensorFlow) across different hardware configurations.**

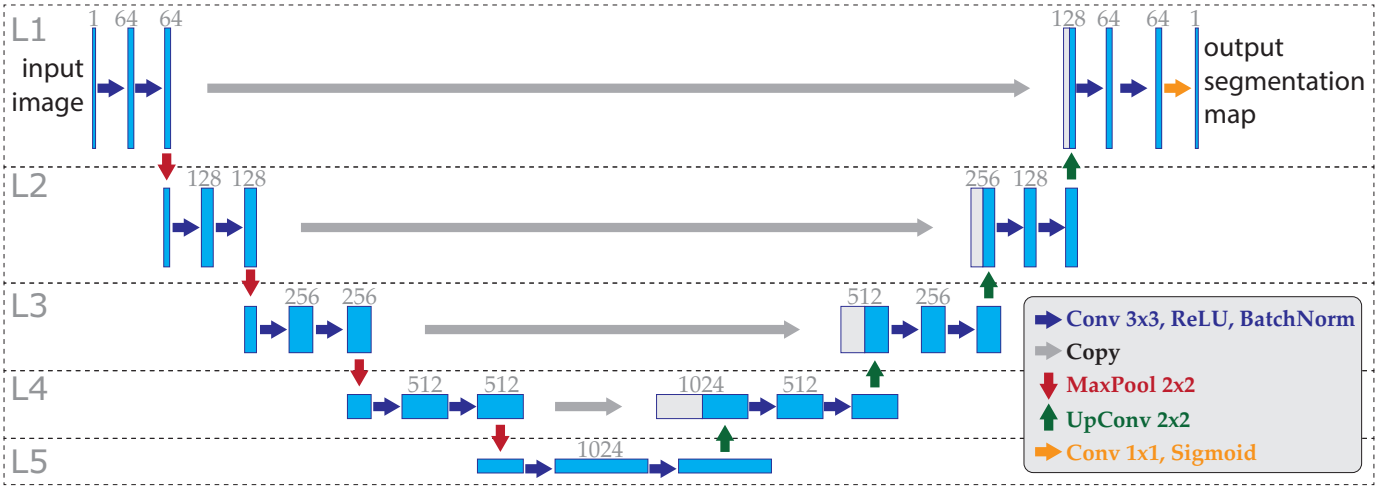
I. Introduction

In the last years, the release of different easy and effective frameworks for Deep Learning (DL) allowed many researchers to explore their applicability in several application contexts. Although this often resulted in state-of-art performance, its use in critical domains starts to raise some concerns [1]. Above all, with the diffusion of DL techniques for biomedical image processing, performing a reliable evaluation of obtained results requires to consider their reproducibility across different hardware and software configurations, both at training and at inference times. This is a non-trivial problem not only because DL training and optimization phases strongly rely on stochastic procedures, but also because of the use of some heuristic considerations (mainly speculative procedures) at training time that, although they help in reducing the required computational effort, tend to introduce non-deterministic behavior, with a direct impact on the results and on the model's reproducibility. Usually, to face this problem, researches take probabilistic considerations about the distribution of data into account or focus their attention on very huge datasets. However, this kind of approach does not fit the medical imaging analysis with Computer-Aided Detection and Diagnosis systems [2], requiring demonstrable proofs of effectiveness and reproducibility. Our opinion is that in this case it is very important to clarify if and to what extent a DL based application is stable and repeatable as well as effective, across different environmental (hardware and software) configurations, in order to determine if the resulting model produces good or bad outcomes just because of luckier or blunter environmental training conditions. Therefore, the aim of this paper is to quantitatively highlight the reproducibility problem of Convolutional Neural Networks (CNN) based approaches for the biomedical image processing, in order to highlight the impact that a given software framework and hardware configuration might have when facing the same problem by the same means.

In particular, we analyze the approach proposed in our previous work [3] for breast tissues segmentation in DCE-MRI by using a modified version of a 2D U-Net CNN [4], a very effective deep architecture for semantic segmentation, using two Deep Learning frameworks (MATLAB and TensorFlow) across different hardware configurations.

978-9-0827-9703-9/19/$31.00 ©2019 IEEE

Fig. 1: Our U-Net proposal for breast tissues segmentation: left side performs the contracting path, right side performs the expansive path.



The rest of the paper is organized as follows: section II introduces the reproducibility issue for the deep learning frameworks and the considered breast segmentation problem; section III reports the obtained producibility results, while section IV draws some conclusions.

II. Reproducibility of Deep Learning Models

Deep Learning is one of the most proliferous topics in the recent years' machine learning research, with applications that go from computer vision (such as natural and biomedical image processing) to automatic speech recognition (ASR) and time-series analysis. On this wave of success, several entities (both industries and academics) released several frameworks to make DL accessible to almost everyone. Although frameworks usually differ on many aspects (used programming language, approach to computation, data processing and storage, etc), they all share the need for advanced General-Purpose GPU (GP-GPU) computing, to be able to handle the huge number of matrix operations made to train a deep neural network.

At the moment of writing this work, NVIDIA is the only provider of a suite of APIs and libraries for Deep Learning, based on their GP-GPU paradigm CUDA, whose capabilities could be well synthesized by cuDNN [5], a GPU-accelerated library of primitives (such as 2D Convolution) for deep neural networks. cuDNN default configuration exploits stochastic and speculative procedures that, although increase the execution speed, introduce uncontrollable factors that can result in not reproducible outcomes. In particular, the following cuDNN routines do not guarantee the reproducibility because they use atomic (i.e. not guaranteed synchronization or ordering constraints for memory operations) functions to speed up the computation: cudnnConvolutionBackwardFilter, cudnnConvolutionBackwardData, cudnnPoolingBackward and cudnnSpatialTfSamplerBackward [5]. The source of this non-reproducibility could be related to some implementation choices of synchronization and kernel verification routines (such as the barrier synchronization) [6]. In all the frameworks using cuDNN (such as MATLAB™, TensorFlow, PyTorch etc.), this causes non-deterministic gradient updates, mainly due to underlying non-deterministic reductions for convolutions (i.e. floating-point operations are not necessarily associative) leading to randomness in the trained models.

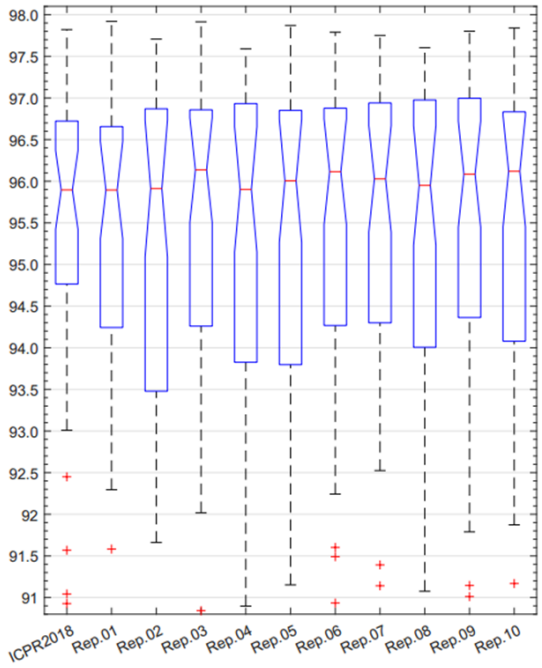
Thus, in this paper, we perform a Montecarlo-like repetition experimentation of our uNet based breast segmentation approach, with the aim of measuring the model robustness and stability over different software frameworks and hardware configurations. To this aim, we performed a Montecarlo-like repetition experimentation, considering the model stable, and thus repeatable, if results stay within a given confidence interval. In our previous work [3] we propose to perform the breast tissues segmentation by considering the 3D volume as a composition of 2D sagittal slices and using a modified 2D U-Net (Figure 1): (a) the output feature-map was set to one to speed up the convergence; (b) zero-padding, with a size-preserving strategy, was applied for preserving the output shapes; (c) batch normalization (BN) layers were inserted after each convolution. The network was trained for 50 epochs by minimizing the task-specific loss $1 - \mathrm{DSC}$, with

$$\mathrm{DSC} = (2 \cdot n(GT \cap SEG))/(n(GT) + n(SEG)) \qquad (1)$$

where $n(\cdot)$ represents the enclosed volume number of voxels, while GT and SEG represents the Ground Truth and produced Segmentation mask respectively.

The network kernel weights have been initialized from a standard distribution [7] $\mathcal{N}(0, \sqrt{2/(\mathrm{fan_i} + \mathrm{fan_o})})$, where $\mathrm{fan_i}$ and $\mathrm{fan_o}$ are the convolution layer input and output features sizes respectively, while the bias weights have been initialized to a constant value of 0.1 to avoid slow-start learning when using ReLu activation functions. ADAM optimizer [8] was

Fig. 2: Boxplots of the ICPR2018 [3] results and of the first 10 out of 50 Montecarlo executions in Table I.



used to minimize the loss function, where we set $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $lr = 0.001$, with an inverse time decay strategy. Performance was evaluated on 42 subjects DCE-MRI data acquired using a 1.5T scanner (Magnetom Symphony, Siemens) equipped with breast coil, considering only the pre-contrast series.
The proposed CNNs have been implemented using two different frameworks:

- **K:** Keras high-level neural networks API in Python 3.6 with the TensorFlow (v1.9) as the back-end
- **M:** MATLAB 2018b with Deep Learning Toolbox 12.0 (formerly Neural Network Toolbox)

Moreover, with the aim of also considering the likely impact of the underlying GPU family, the nets have been evaluated on the following hardware configurations:

- **Conf. A:** A virtual environment freely offered by Google Colaboratory (https://colab.research.google.com). The virtual machine has an Intel(R) Xeon(R) @ 2.2GHz CPU (2 cores), 13GB RAM and an Nvidia K80 GPU (Tesla family) with 12GB GRAM (Tested framework: K)
- **Conf. B** A physical server hosted in our university HPC center (http://www.scope.unina.it) equipped with 2 x Intel(R) Xeon(R) Intel(R) 2.13GHz CPUs (4 cores), 32GB RAM and an Nvidia Titan Xp GPU (Pascal family) with 12GB GRAM (Tested frameworks: K and M)
- **Conf. C** A DELL R720 equipped with two Intel(R) Xeon(R) CPU E5-2680 v2 @ 2.80GHz, 128GB RAM and two NVIDIA Tesla K20 (Pascal family) with 5GB GRAM (Tested frameworks: K and M)

TABLE I: Results obtained for each of the first 10 out of 50 Montecarlo executions of the 10-fold cross-validation for our approach, using the **Conf. A** and the Framework **K**. The results presented in ICPR2018 [3] are also reported in bold. Median values with corresponding 95% confidence intervals (LB: LowerBound, UB: UpperBound) are reported.

Repetition	DSC [%]	LB [%]	UB [%]
ICPR2018 [3]	**95.90%**	**95.16%**	**96.64%**
Rep.01	95.80%	95.24%	96.37%
Rep.02	96.19%	95.62%	96.75%
Rep.03	95.85%	95.38%	96.39%
Rep.04	96.11%	95.69%	96.57%
Rep.05	96.04%	95.15%	96.62%
Rep.06	95.90%	95.02%	96.60%
Rep.07	96.25%	95.29%	96.52%
Rep.08	95.93%	95.44%	96.56%
Rep.09	95.95%	95.38%	96.36%
Rep.10	95.89%	95.35%	96.43%

The assessment was performed by using a patient-based 10-fold Cross Validation (CV), in order to prevent slices from the same subject belonging to two different folds, applying a training/test data standardization using the median and standard deviation calculated only on the training patients' fold. To validate the repeatability of our model, we repeated the execution 50 times. We used the same initialization seeds for the random numbers generators to try highlighting only the uncertainty due to random considerations introduced by the optimization tools' randomness, minimizing the effects that could be raised by a different source of randomness. The obtained breast-mask is compared to the gold standard in terms of Dice Similarity Coefficient (DSC) index.

III. Results

For brevity reasons, this section tables reports only the first 10 executions of the Montecarlo analysis for each configuration and only the boxplot related to table 1 results. Each Montecarlo execution applies a 10-fold cross-validation producing 10 folds containing the 42 patient segmentations. The median values of a part of the Montecarlo execution are reported in the Tables I, II, III, IV and V.

Tables I to V show how the computational frameworks for the optimization of deep learning models suffer from reproducibility during the training phase producing different models and thus, different results. This problem is not limited to the analyzed framework, MATLAB and TensorFlow, nor

TABLE II: Results obtained for each of the first 10 out of 50 Montecarlo executions of the 10-fold cross-validation for our approach, using the **Conf. B** and the Framework **K**. The results presented in ICPR2018 [3] are also reported in bold. Median values with corresponding 95% confidence intervals (LB: LowerBound, UB: UpperBound) are reported.

Repetition	DSC [%]	LB [%]	UB [%]
ICPR2018 [3]	95.90%	95.16%	96.64%
Rep.01	95.89%	95.18%	96.47%
Rep.02	95.91%	95.25%	96.32%
Rep.03	96.14%	95.08%	96.66%
Rep.04	95.90%	94.92%	96.48%
Rep.05	96.01%	94.98%	96.41%
Rep.06	96.12%	94.95%	96.53%
Rep.07	96.03%	95.56%	96.28%
Rep.08	95.95%	95.52%	96.29%
Rep.09	96.08%	94.77%	96.39%
Rep.10	96.12%	95.31%	96.48%

TABLE III: Results obtained for each of the first 10 out of 50 Montecarlo executions of the 10-fold cross-validation for our approach, using the **Conf. B** and the Framework **M**. The results presented in ICPR2018 [3] are also reported in bold. Median values with corresponding 95% confidence intervals (LB: LowerBound, UB: UpperBound) are reported.

Repetition	DSC [%]	LB [%]	UB [%]
ICPR2018 [3]	95.90%	95.16%	96.64%
Rep.01	96.25%	95.43%	96.53%
Rep.02	95.86%	95.40%	96.15%
Rep.03	95.86%	94.94%	96.08%
Rep.04	96.11%	95.61%	96.52%
Rep.05	95.99%	95.27%	96.27%
Rep.06	95.90%	95.15%	96.26%
Rep.07	95.91%	95.22%	96.31%
Rep.08	96.21%	95.64%	96.46%
Rep.09	95.95%	95.62%	96.13%
Rep.10	95.94%	95.64%	96.18%

it depends on the used GPU architecture, but it lies in the Nvidia libraries as discussed in Section II. Nevertheless, the randomness introduced in the trained models (by fixing the seeds of all the random numbers generators) produces not statistically different results as graphically shown in Figure 2. Finally, Table VI reports the statistics (median values) about confidence intervals (CIs) and training times for each of the experiments to better compare and discuss the results. The CI size has been calculated as the difference between the Upper Bound and the Lowe Bound (UB - LB).

TABLE IV: Results obtained for each of the first 10 out of 50 Montecarlo executions of the 10-fold cross-validation for our approach, using the **Conf. C** and the Framework **K**. The results presented in ICPR2018 [3] are also reported in bold. Median values with corresponding 95% confidence intervals (LB: LowerBound, UB: UpperBound) are reported.

Repetition	DSC [%]	LB [%]	UB [%]
ICPR2018 [3]	95.90%	95.16%	96.64%
Rep.01	96.05%	94.87%	96.33%
Rep.02	95.99%	95.11%	96.51%
Rep.03	96.05%	95.32%	96.38%
Rep.04	96.00%	95.61%	96.30%
Rep.05	95.91%	94.98%	96.27%
Rep.06	96.04%	95.09%	96.51%
Rep.07	96.14%	95.08%	96.50%
Rep.08	95.99%	95.35%	96.51%
Rep.09	95.92%	95.32%	96.28%
Rep.10	96.10%	95.68%	96.32%

TABLE V: Results obtained for each of the first 10 out of 50 Montecarlo executions of the 10-fold cross-validation for our approach, using the **Conf. C** and the Framework **M**. The results presented in ICPR2018 [3] are also reported in bold. Median values with corresponding 95% confidence intervals (LB: LowerBound, UB: UpperBound) are reported.

Repetition	DSC [%]	LB [%]	UB [%]
ICPR2018 [3]	95.90%	95.16%	96.64%
Rep.01	95.98%	95.66%	96.15%
Rep.02	95.92%	95.20%	96.19%
Rep.03	96.16%	95.67%	96.56%
Rep.04	95.84%	95.38%	96.13%
Rep.05	95.88%	95.15%	96.25%
Rep.06	95.91%	95.00%	96.13%
Rep.07	96.19%	95.63%	96.44%
Rep.08	95.86%	95.56%	96.09%
Rep.09	96.19%	95.38%	96.48%
Rep.10	95.95%	95.27%	96.36%

IV. Discussions and Conclusions

The aim of this paper was to quantitatively highlight the problem of the reproducibility of Deep Learning based approaches for biomedical image processing, in order to highlight the impact that a given software framework and hardware configurations might have when facing the same problem by the same means.

Although results show that the reproducibility problem exists, it is worth noticing that it is not limited to the analyzed frameworks (MATLAB and TensorFlow), neither in the used GPU architecture, but it lies in the Nvidia libraries as discussed in Section 2 (and this is further confirmed by the fact that several CPUs executions produce totally reproducible results).

TABLE VI: Comparative results (median values) for each experiment set-up.

Conf.	**A**	**B**	**B**	**C**	**C**
Framework	**K**	**K**	**M**	**K**	**M**
Median CI size	1.13%	1.36%	0.95%	1.22%	0.94%
Median training time	∼50min	∼13min	∼33hours	∼25min	∼42hours

Moreover, tables I to V show that the variability across different frameworks is more evident than the variability across different hardware architectures.

Analyzing the boxplot in Figure 2, we can state that our CNN-based model is stable to the different training executions over different frameworks and hardware configurations (since the confidence intervals obtained on the tests data overlap). It is interesting to note that, although from a statistical point of view there are no significant differences among the configurations (both hardware and software), the model trained with MATLAB appears to be more stable, since its confidence intervals are narrower (about 27% smaller). This suggests that the MATLAB framework better compensates for the randomness associated with the training, paying it in terms of training time.

Generally speaking, the randomness introduced by deep learning libraries, could impact outcomes of biomedical image processing application relying on deep learning approaches. Therefore, in order to not provide not totally reproducible claims, it is very important to shifts the attention from a pure performance point-of-view to a statistical reproducibility of the obtained models, since a model showing large variations in results will have wider confidence intervals with respect to a more stable one.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research, the availability of the Calculation Centre SCoPE of the University of Naples Federico II and thank the SCoPE academic staff for the given support. The authors are also grateful to Dr. Antonella Petrillo, Head of Division of Radiology and PhD Roberta Fusco, Department of Diagnostic Imaging, Radiant and Metabolic Therapy, "Istituto Nazionale dei Tumori Fondazione G. Pascale" - IRCCS, Naples, Italy, for providing data. This work is part of "Synergy-net: Research and Digital Solutions against Cancer" project (funded in the framework of the POR Campania FESR 2014-2020).

REFERENCES

[1] G. Ras, M. van Gerven, and P. Haselager, "Explanation methods in deep learning: Users, values, concerns and challenges," in *Explainable and Interpretable Models in Computer Vision and Machine Learning*. Springer, 2018, pp. 19–36.

[2] G. Piantadosi, S. Marrone, R. Fusco, M. Sansone, and C. Sansone, "Comprehensive computer-aided diagnosis for breast t1-weighted dce-mri through quantitative dynamical features and spatio-temporal local binary patterns," *IET Computer Vision*, vol. 12, no. 7, pp. 1007–1017, 2018.

[3] G. Piantadosi, M. Sansone, and C. Sansone, "Breast segmentation in mri via u-net deep convolutional neural networks," in *2018 24th International Conference on Pattern Recognition (ICPR)*. IEEE, 2018, pp. 3917–3922.

[4] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[5] S. Chetlur, C. Woolley, P. Vandermersch, J. Cohen, J. Tran, B. Catanzaro, and E. Shelhamer, "cudnn: Efficient primitives for deep learning," *arXiv preprint arXiv:1410.0759*, 2014.

[6] E. Bardsley and A. F. Donaldson, "Warps and atomics: Beyond barrier synchronization in the verification of gpu kernels," in *NASA Formal Methods Symposium*. Springer, 2014, pp. 230–245.

[7] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[8] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.