# Exploring Different Convolutional Neural Networks Architectures to Identify Cells in Spheroids

A.G. Santiago[1], C.S.Campos[1], M. M. G. Macedo[2], J.K.M.B. Daguano[1,3], J.A. Dernowsek[3,4] and A.C.D. Rodas[1]

[1] Federal University of ABC/Center for Modeling, Engineering and Social Sciences,São Bernardo do Campo, Brazil
[2] IBM Research, Sao Paulo, Brazil
[3] CTI - Renato Archer, Campinas, Brazil
[4] Institute of Energetics and Nuclear Research, Biotechnology Center, São Paulo, Brazil

*Abstract*— **The cultivation of cells in 3D has gained more interest in research once 3D architecture can be closer to full cell physiological functionality. The cultivation of the cells in a spheroid format has shown very promising results, further for bioprinting developing so fast during the last decade. The interaction of spheroids and the matrix, or bioink, have proportionate new structures to be analyzed, specially if one would like to follow the whole system (spheroid and bioink) without fluorescent dyes. Trying to solve this image limitation, the aim of this paper is to present a study on different Convolutional Neural Networks (CNN) architectures employed to identify different structures in fibroblast NIH-3T3 spheroids. Three different architectures were considered: GoogleNet, ResNet18 and AlexNet, all implemented in Python 3.7 using the PyTorch Application Interface Programming (API). Given a spheroid image taken in a light microscope, four structures can be identified: the cell, the dead cell, the impurity/contamination and the background consisting of a gel in which the spheroid is immersed. All four CNN architectures were trained and evaluated with a dataset consisting of over 370 samples, split into a training set ($\approx 70\%$), a test set ($\approx 20\%$) and a validation set ($\approx 10\%$). Since our dataset has unbalanced classes, a data augmentation was applied in order to provide a comparable number of samples for all classes being considered.**

*Keywords*— **Cell spheroids, Fibroblast NIH-3T3, CNN architectures, Cell analyses, Digital Process Automation**

## I. Introduction

Cellular spheroid is a complex aggregate of cells, non-substrate-adherent in a spherical shape, composed from hundred up to thousands of mutually adherent populations of cells [1]. Imaging a spheroid is an important step in order to understand how cell behavior operates and give references about possibilities according to the cell type of spheroid formation. As spheroids have an easy handling and suitability for high quality imaging, doing such analysis provides a great representation of three dimensional organization of cells which, in this case, is much similar to real tissue when compared to a 2D monolayer cell culture [2]. Cells cultivated in spheroid are getting more attention once this kind of structure can mimic the 3D architecture for the cell. Studies using a combination of spheroids has shown results closer to the functional tissue [3, 4] and bioprinting targeting the regenerative medicine [5]. The studies using spheroids to get the functional results take time and it is important to follow the transformation of the spheroids with non invasive techniques and simple equipment as light microscopy. For this reason, the objective of this work is, based on the work of Rodrigues [6], verify which Convolutional Neural Network (CNN) could be applied to start the observation of the viable cells in the spheroids.

## II. Methodolology

### A. Spheroid images

Spheroids were prepared by non-adhesive hydrogel technique. Basically, a NIH-3T3 fibroblasts suspension with $1 \times 10^5$ $cells/mL$ were poured in a well from a 12 well plate previously covered with agarose and molded micro vessels with a mold developed at CTI-Renato Archer. The cells were left in an incubator at $37^oC$ at the humidity chamber with 5% increase of $CO_2$ for 3 days, when they were pictured in a contrast phase microscope with 200 times magnification.

### B. Dataset Preparation

The image dataset consists of 370 samples of cell image segmented from two spheroids images with impurity (8.33%), live NIH-3T3 cells (8.33%), dead cells (41.66%) and background (41.66%). Figure 1 presents the classification criteria, which the samples was split into train, test and validation datasets corresponding to 70%, 20% and 10% of the original data respectively. Since each class presents a considerable difference in the number of samples ($5\times$ in

some cases), data augmentation techniques were applied to the training set in order to perform equalization of samples, thus, improving the quality of the training process.

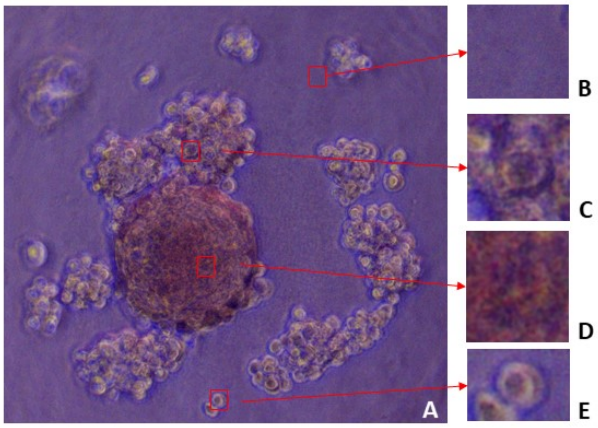



Fig. 1: Spheroid of the fibroblast cell line NIH-3T3. a) Caption of the whole spheroid; b) Background; c) Live cell that does not belong to the spheroid, d) Live cell that belongs to the spheroid; e) Dead cell

The transformations applied are given below:

- Random rotation with $-10^o \leq \theta \leq 10^o$;
- Random horizontal flip with a probability of 50%;
- Random vertical flip with a probability of 50%;

After the data augmentation procedure, a new training dataset was generated with background (24.57%), impurity (25.71%), live NIH-3T3 cell (25.71%) and dead cells (24%).

In order to improve the CNN performance, another set of transformations were applied to both, train and test datasets:

- Image resize to $260 \times 260$ pixels;
- Color channel normalization;

The image resize step is needed in order to allow the proper use of the desired CNN architectures, such as GoogLeNet, which requires a minimum image size of $260 \times 260$ pixels.

For the color channel normalization, all pixel values were normalized between $[0,1]$ by the PyTorch Tensor object [7] using a linear function and followed by a second normalization according to the Equation 1:

$$I_{ij} = \frac{I_{ij} - \mu_c}{\sigma_c} \tag{1}$$

where the subscript c corresponds to a given color channel and $\mu_c$ and $\sigma_c$ are the mean and standard values respectively. For each channel, $\mu_c$ and $\sigma_c$ are given by Table 1:

Table 1: Mean and standard deviation for a given channel

| Channel | $\mu_c$ | $\sigma_c$ |
|---|---|---|
| 1 | 0.485 | 0.229 |
| 2 | 0.456 | 0.224 |
| 3 | 0.406 | 0.225 |

By applying Equation 1, the values are normalized to the range $[-1,1]$ with zero-mean, improving the training and testing process by bounding the neural network weight values [8].

### C. CNN Implementation

All CNNs architectures were implemented using Python 3.7 Anaconda [9] distribution and the PyTorch API. Within the train and test datasets, a class corresponding folder was created and labeled automatically using the torch ImageFolder method. The hardware specification is a core i7 with 16Gb and a Tesla K80 Nvidia graphic processor with 24Gb of RAM. Were considered the following CNN architectures:

- AlexNet [10];
- GoogLeNet [11];
- ResNet 18 [12];

For each architecture, the classifying layer was altered in order to perform the classification of the four different structures considered, as shown in Table 2.

In Table 2, for the Dropout layer, the variable $p$ corresponds to the probability of certain neurons being randomly deactivated during the training process.

### D. Training and Testing Process

In order to proceed with the training process, each class was labeled from 0 to 3. The *Adam* optimizer was chosen for all CNN architectures along the *Cross-Entropy* (CE) given by Equation 2.

$$CE = \sum_{i=0}^{N_{class}} y_i log(y_i - 1) + (y_i - 1) log(y_i) \tag{2}$$

In Equation 2, $y_i$ is the predicted value for a given sample.

Different hyperparameters sets consisting of the number of epochs for training, learning rate and training batch sizes were considered and are given by Table 3.

Table 2: Classifying layers for the architectures considered

| Architecture | Classifying layer |
|---|---|
| AlexNet | Dropout (p=0.5) |
| | Linear (input features = 9216, output features = 4096) |
| | Rectifier Linear Unit |
| | Dropout (p=0.5) |
| | Linear (input features = 4096, output features = 4096) |
| | Rectifier Linear Unit |
| | Linear (input features = 4096, output features = 4) |
| | Logarithmic SoftMax |
| GoogLeNet | Rectifier Linear Unit |
| | Linear (input features = 1024, output features = 4) |
| | Logarithmic SoftMax |
| ResNet 18 | Linear (input features = 512, output features = 4) |
| | Logarithmic SoftMax |

Table 3: Hyperparameters used for training the CNNs

| Set | Epochs | Batch size | lr |
|---|---|---|---|
| 1 | 50 | 20 | $1.0 \times 10^{-3}$ |
| 2 | 50 | 20 | $1.0 \times 10^{-4}$ |
| 3 | 50 | 40 | $1.0 \times 10^{-3}$ |
| 4 | 50 | 40 | $1.0 \times 10^{-4}$ |
| 5 | 100 | 20 | $1.0 \times 10^{-3}$ |
| 6 | 100 | 20 | $1.0 \times 10^{-4}$ |
| 7 | 100 | 40 | $1.0 \times 10^{-3}$ |
| 8 | 100 | 40 | $1.0 \times 10^{-4}$ |

Table 4: Trainable parameters for each CNN architecture

| CNN Architecture | Number of Parameters |
|---|---|
| AlexNet | 57020228 |
| ResNet 18 | 11178564 |
| GoogLeNet | 5604004 |

The testing loss and accuracy were computed along with the training procedure at the end of each batch loop with the test batch size fixed at 10 samples. For each CNN architecture, the number of trainable parameters is presented in Table 4.

## III. Results

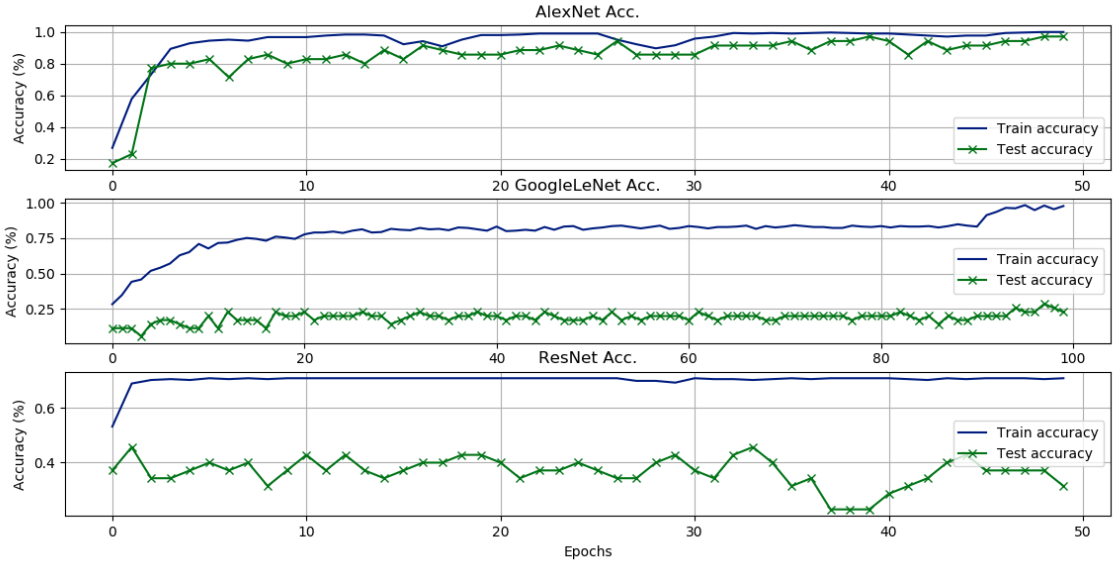
Each CNN architecture was trained and tested according to the parameters described. The corresponding training and testing loss and accuracy are presented in Figures 2(a) and 2(b) respectively, considering AlexNet (hyperparameters set 1); GoogLeNet (set 6); ResNet 18 (set 2). All other hyperparameters sets presented lower accuracy rate and higher loss values.

The computational time needed to perform the training/ testing procedure was evaluated and it is presented in Table 5 for the corresponding plots in Figure 2.

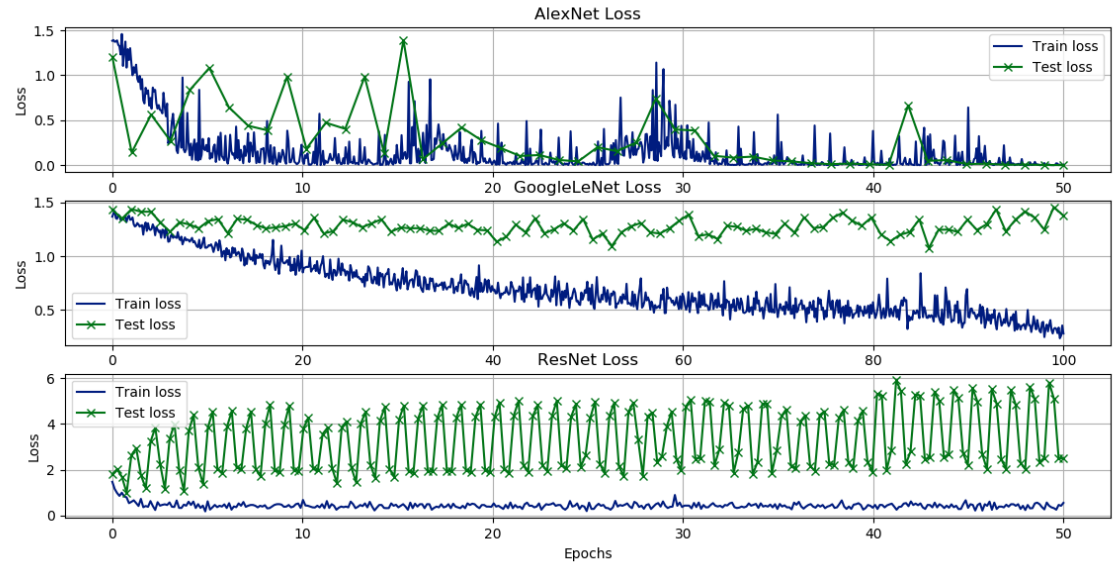
Figure 3 presents the confusion matrix for AlexNet. The confusion matrix was evaluated considering the validation dataset, which consists of 35 samples with 23 background images (label 0), 2 dead cell images (label 1), 2 living cell outside spheroid images (label 2) and 8 living cell inside spheroid images (label 3).

## IV. Discussions

As can be observed in Figure 2(a), all architectures presented a decaying tendency for the training set loss function, with AlexNet also presenting oscillating values higher than GoogLeNet and ResNet 18. On the other hand, for the testing set, Figure 2(b), AlexNet presented a decaying behavior as the number of training epochs increased but still with oscillating values. GoogLeNet and ResNet presented an oscillating behavior with no decaying tendency. It must be noted that GoogLeNet still has room for improving its training loss function curve with a higher number of epochs. It is important to note that the behavior observed for all architectures is consistent with those found at Rodrigues work [6]. Figure 2(b) shows that the CNN architecture with the highest accuracy for the test set is the AlexNet, with a success rate of almost 98.8% and with a training time of 26.54 minutes, while GoogLeNet presents an accuracy for the test set of $\approx 25\%$ with a tendency to remain constant and ResNet presented an accuracy of, at most, $\approx 40\%$ but oscillating. It is important to notice that, according to Figure 3, the AlexNet architecture was able to identify 100% of the living cells inside the spheroid (label 3), which indicates that this model can be used to evaluate the number of those cells as a non-




(a)




(b)

Fig. 2: CNN training and testing. a) Loss for AlexNet; GoogLeNet and ResNet 18. b) Accuracy for AlexNet; GoogLeNet and ResNet 18. The green line is the test set and the blue line the training set

Table 5: Computational time for training/ test corresponding to Figure 2

| CNN Architecture | Comp. Time [min] |
|---|---|
| AlexNet | 26.54 |
| ResNet 18 | 56.50 |
| GoogLeNet | 116.20 |

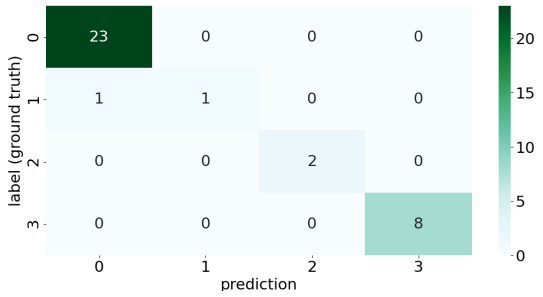



Fig. 3: AlexNet confusion matrix for the validation dataset: 0 - Background; 1 - Dead Cell; 2 - Living cell outside the spheroid; 3 - Cell inside spheroid

invasive procedure. One important point to have a CNN for this kind of identification is the possibility of following the spheroids behaviour when cultured in more complex structure. The identification of single cells in these models usually are handmade, which requires a highly qualified person and usually the analysis criteria is not the same from one person to another, making it time consuming. The preparation of the Figure 1 brought a lot of discussion among the authors and the time to do the segmentation to train the CNN took about 40 minutes.

## V. Conclusions

This work wanted to explore different CNN for cell identification in spheroid structures. Here, we started with a simple structure to do this exploration and the AlexNet showed to be the promising one. As next steps, this CNN is intended to be used in more complex images.

## Acknowledgments

The authors would like to thank CAPES for financial support – CODE 001.

## Conflict of Interest

The authors declare that they have no conflict of interest.

## References

1. Pampaloni F., Stelzer E. H. K.. Three-dimensional cell cultures in toxicology *Biotechnology and Genetic Engineering Reviews.* 2009;26:117–138.
2. Fischer C. S.. An Introduction to Image-Based Systems Biology of Multicellular Spheroids for Experimentalists and Theoreticians *Computational Biology.* 2019:1–18.
3. Chiew G. G. Y., Wei N., Sultania S., Lim S., Luo K. Q.. Bioengineered three-dimensional co-culture of cancer cells and endothelial cells: A model system for dual analysis of tumor growth and angiogenesis *Biotechnology and Bioengineering.* 2017;114:1865–1877.
4. Klimkiewicz K., Weglarczyk K., Collet G., et al. A 3D model of tumour angiogenic microenvironment to monitor hypoxia effects on cell interactions and cancer stem cell selection *Cancer Letters.* 2017;396:10–20.
5. Miri Amir K, Khalilpour Akbar, Cecen Berivan, Maharjan Sushila, Shin Su Ryon, Khademhosseini Ali. Multiscale bioprinting of vascularized models *Biomaterials.* 2019;198:204–216.
6. Rodrigues Larissa Ferreira, Naldi Murilo Coelho, Mari Joao Fernando. Exploiting Convolutional Neural Networks and Preprocessing Techniques for HEp-2 Cell Classification in Immunofluorescence Images *Proceedings - 30th Conference on Graphics, Patterns and Images, SIBGRAPI 2017.* 2017:170–177.
7. PyTorch https://pytorch.org/docs/stable/torchvision/transforms.html. Accessed: 2020-03-27.
8. Csink L., Paulus D., Ahlrichs U., B. Heigl B. Color Normalization and Object Localization *Review Literature And Arts Of The Americas.* 1998;6.
9. Anaconda 3.7 https://www.anaconda.com/products/individualm. Accessed: 2020-03-27.
10. Krizhevsky A., Sutskever I., Hinton G. E.. ImageNet Classification with Deep Convolutional Neural Networks *Advances in neural information processing systems.* 2012.
11. Szegedy C., Liu W., Jia Y., et al. Going Deeper with Convolutions in *Proceedings of the IEEE conference on computer vision and pattern recognition*:1-9 2015.
12. He K., Zhang X., Ren Shaoqing, Sun J.. Deep residual learning for image recognition in *Proceedings of the IEEE conference on computer vision and pattern recognition*:770-778 2016.

Author: Andrea Cecilia Dorion Rodas
Institute: Center for Engeneering, Modeling and Applied Social Sciences
Street: Alameda da Universidade s/n
City: São Bernardo do Campo
Country: Brazil
Email: andrea.rodas@ufabc.edu.br