Paper ML01

Using Image recognition in production support

Thomas Dekelver, Business & Decision Life Sciences, Brussels, Belgium

ABSTRACT

In the production of a vaccine, multiple steps are required. In this case, the antigens are grown on petri dishes, where at a certain stage of the production a quality check (QC) must be performed through a visual inspection to check the sterility of the production zones. This can be seen visually as the presence or absence of black spots, the Colony Forming Units (CFU), in the petri dish. When CFUs are detected, the petri dish is contaminated and must be discarded. This is a routine, time-consuming, error-prone, and still very labour-intensive work, nevertheless essential step in the production process. We have developed an image-recognition approach, based on the latest developments in AI and deep-learning, to automate the process and hence increase operational efficiency. In this paper, we will show how to use these techniques and discuss about the pitfalls encountered during the development, the lessons learned and provide some recommendations.

INTRODUCTION

Still many tasks humans are performing are very repetitive and consequently, high volume, time consuming and error prone. Nevertheless, these are usually essential tasks in a production process. This is also the case in the production of a vaccine. Multiple steps are required and one of these steps is the quality check (QC). Antigens are grown on petri dishes and a visual inspection is needed to check the sterility of production zones. Concretely, this is done by operators that check the absence or presence of black spots, the Colony Forming Units (CFU), or other anomalies, such as a corrupted agar. In some cases, thousands of petri dishes are produced every day. Consequently, automatization of this task would result in a big advantage, by a) reducing the amount of repetitive and error-prone work for the operators, giving them more time for other tasks, b) increasing the reliability and consistency of this task.

The automatization of bacterial counting has been covered in a lot of scientific papers in the last decades as it is a classical challenge in many Clinical Microbiology Laboratories [1]. However, this was usually automated using “classical” machine-learning techniques, which bring with them a set of limitations. Thanks to the recent developments in computational power, together with a greater availability of large datasets, there has been a huge boost in Artificial Intelligence (AI) applications, and particularly in computer vision [2]. With the help of deep neural networks, we can address many of the challenges encountered in the traditional approach.

This paper has been targeted to readers unfamiliar with the machine learning world. Consequently, if you haven’t been much in contact with machine learning, you should still be able to understand the topics discussed below. If you are a more experienced reader and you want more details, please contact the author for further questions. The paper will first discuss the difference between “classical” machine learning and deep learning, specifically in image classification and for bacterial counting. Secondly, we will explain the task and data used in this project, followed by the modelling process we established, with its multiple iterations. Finally, we will discuss our results, together with some visualisations.

CLASSICAL MACHINE LEARNING VS. DEEP LEARNING IN IMAGE CLASSIFICATION

Image classification is a task in machine learning that tries to automatically extract information from images. The goal is to classify the image in one of two (2) (binary classification, i.e. is this image a cat or a dog?) or more (multiclass classification, i.e. what is the breed of the dog in this picture?) predefined classes. Currently there exist two big groups of machine learning techniques that can perform image classification: “classical” machine learning and deep learning.

Three to five years ago it was not common to use deep learning in an image classification task. Or at least it was not very performant. This has changed a lot the last few years with developments in computational power (dedicated faster processors [Graphics Processing Units], affordable cloud services, etc.), together with a greater availability of large datasets [2]. Thanks to this, big advancements in the algorithms themselves were made. In the two sections below, we will detail the differences of both methods, as well as their respective advantages and disadvantages.

CLASSICAL MACHINE LEARNING

Machine learning is a sub-field of AI and is based on algorithms that can learn from data without relying on rules-based programming [3]. This is the traditional approach used. It uses classical machine learning algorithms (such as Support Vector Machines (SVM) or Classification Trees). However, these algorithms need numerical vectors as an input, to be

able to make a prediction. This means that images need to be transformed. An image is essentially a triple (3 colours, RGB) matrix of pixel values ranging from 0 to 255. This is called the feature extraction step. The "features" or variables from the image need to be manually created, to populate the input vector to use in the algorithm. This is a very labour-intensive step. It is very complex, as it is not easy to know what feature will have a high predictive power (i.e. help a lot in making a good prediction). In addition, it requires a lot of manual work and experience to select and create these features with high predictive power [4].

Specifically, in the area of bacterial counting, some researchers were able to achieve performances comparable to human technicians, but with the following limitations [5-17]:

a) Those performances were met only in very specific scientific settings (only a few strains, limited amount of plates, …).
b) Most methods are parametric, which makes them not scalable.
c) Many still require strong need of human presence even after the modelling process.

Consequently, it remains still hard to perform some automated quality checks based on "classic" machine learning.

DEEP LEARING

Deep learning is a sub-field of Machine Learning that is based on neural networks. These are called deep neural networks, as they are usually composed of a large number of layers, where the input of one layer is the output of the previous layer. It is a technique that exists already since the 1950s, but that only recently, showed big enhancements in predictive power in, among other things, image classification. Along with state-of-the-art performances in image classification, deep learning also helps to address some of the challenges we encountered with "classical" machine learning. Firstly, no feature extraction is needed. Where in classical machine learning, this was a very lengthy and complicated process, in deep learning, we directly feed the image as an input for the algorithm, without a need to create features manually. Secondly, there is a theoretical theorem (the Universal approximation theorem) that states: "A neural network can solve any given problem to arbitrarily close accuracy as long as you add enough parameters" [18]. Although being not that straightforward practically, this theorem still confirms that we can achieve very high performances with neural networks. Thirdly, neural networks have improved generalisation properties compared to "classical" machine learning algorithms. This means that if you train the algorithm on a given dataset, it will have a greater ability to give good results on a similar dataset from a different source or with slight variations.

In deep learning, a specific neural network has been designed to handle image classification. It is called a Convolutional Neural Network (CNN) and is based on the spatial structure of images. The name comes from the term convolution. It can be described as a matrix multiplication between a filter (weight matrix) and a square of pixels of the same size in the image. A CNN uses 2 properties to reduce the number of parameters to optimize, which makes it easier to optimise. Firstly, thanks to the local connectivity property, pixels close to each other are treated together as they influence each other. Secondly, with the parameter sharing property, parameters are shared over the image as a given pattern could happen anywhere in the image. Indeed, no distinction should be made if the cat face appears in the top right or bottom left part of the image, it is still the same cat.

Finally, linked to the better generalisation properties, an important technique has made the training of neural networks easier and more efficient. This technique is called transfer learning. When you use transfer learning, you start from a pre-trained network, i.e. a network that has already been trained on another image dataset, and you cut off the last part of the network, to replace it with custom layers of your problem. These pre-trained networks are usually trained on the ImageNet dataset [19], consisting of 14 million images, classified in 20,000 classes. Consequently, the last layer of the network is going to make a prediction on 20,000 classes. However, in our case we only want to predict if the antigens growing on the petri dish are contaminated or not (binary decision, 2 classes). Therefore, we need to chop off the last layer of the network and replace it with a few (usually 3) custom layers that give a prediction on 2 classes, instead of the 20,000 classes. This is a very surprising technique, as you first learn the model to recognize 20,000 classes (of which for instance food, birds, plants, …) to then modify it to give a prediction on only 2 classes (contaminated or not). But if you take a closer look at what the model learns in each layer, you can understand why this works so well. In the first few layers, the neural network learns to recognize simple colours and lines. After a few layers, the network is able to recognize simple shapes such as circles and squares. The further you go in the layers, the more fine-grained and complex patterns the network is able to identify. Consequently, it seems logic that the first set of layers is needed in any image classification problem, as you would always need to know what a line or a colour is.

THE TASK AND DATA

As described in the introduction, we want to be able to automate a quality check of the production of vaccines, by building a supervised deep learning model (supervised because we give labels to each image and the model learns from the examples that we feed him). Therefore, we need to assess the sterility of the petri dishes. In our specific case, we are working with a pharmaceutical producer that uses two different sizes of petri dishes: 55 mm contact plates and 90 mm settle plates. During the QC, the plates are fed into a robotic machine that takes a plate, removes the lid, takes a picture, puts back the lid and places the plate in the correct pile (contaminated vs non-contaminated). Subsequently,

the plates that were placed in the contaminated pile will be rechecked by an operator, while the non-contaminated plates will automatically go to the next step in the process. Therefore, it is crucial to have as few as possible false negatives (plates that are predicted as non-contaminated, but actually are contaminated).

We also faced some challenges. Actually, the operators are asked to put their annotations (marker & tape) as well as the unique tag to identify the plate on the top lid of the plate, so that the picture taken is not influenced by those. However, in reality this doesn't always happen and we have collected multiple pictures with the tag, marker annotations, tape and/or glue on the bottom lid of the plate. In addition, on some plates the agar is corrupted, which is considered as also being contaminated and a technical operator should asses if the plate can go to the next stage of the production or not. Moreover, some plates contain air bubbles and/or particles. These are considered to be non-contaminated plates. Consequently, we had to incorporate these "anomalies" in the model building phase.

We used a dataset consisting of 5,771 labelled images (3,016 images of 55 mm plates and 2,755 images of 90 mm plates). These were shuffled and divided in 3 sets: 70% in the training dataset (used for the training of the model), 15% in the validation dataset (used to validate the hyperparameters of the model) and 15% in the testing dataset (used to assess the model performances). In addition, 1,083 unlabelled images were used as second testing set to assess the model performances.

THE MODELLING PROCESS

We followed an iterative (agile) modelling process, similar to the one described in the CRISP-DM methodology [20], as shown in figure 1. First, we made sure the images were labelled correctly. Second, we pre-processed the images. Third, came the actual model training phase, which needed as input some hyperparameters (these are parameters with whom you can fine-tune the model to slightly change its behaviour and, consequently, its results). Finally, we had some results that had to be evaluated and inspected, to see where we could improve and perform an additional iteration. This was done in continuous collaboration with the business stakeholders, to make sure we were going in the right direction and we understood the business problem correctly.

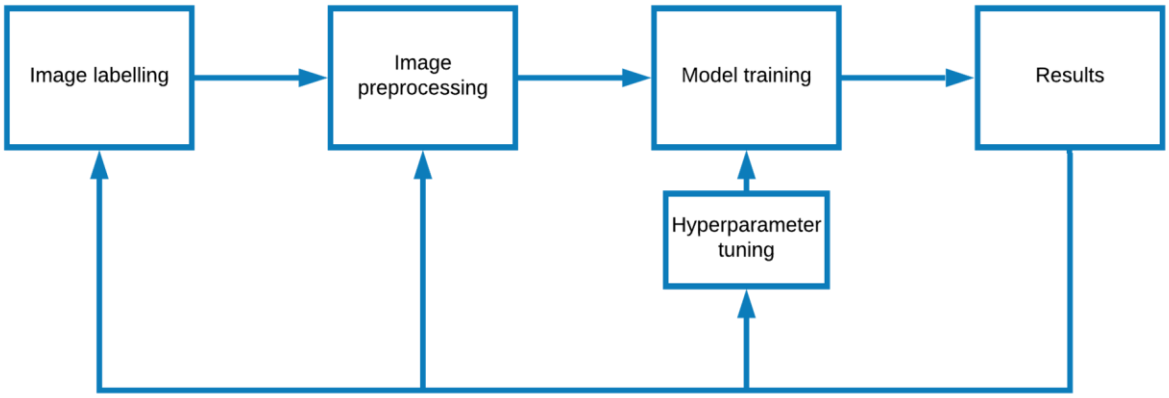


Figure 1: Iterative (agile) process used to build the model with the highest performance possible

The first steps to create a deep learning model are to select a few starting parameters. This part will be a bit more technical, but it is not mandatory to understand the rest of the topics discussed. We used a deep learning framework called fast.ai, that is built on top of PyTorch and implemented in python. As explained above, no feature extraction from the images to use it with a CNN are needed. However, some hyper-parameters need to be defined. We resized the images from 2046 x 2046 pixels to 224 x 224 pixels, which is a standard size used by most deep learning researchers. Indeed, if the full image would be fed to the network, it would require too much computational power before getting any satisfactory results. Next, it needs to be chosen which (pre-trained) model will be used and what size (i.e. how many layers). Here we started with ResNet50, as it has the best scores on the DawnBench [21]. Finally, it needs to be defined what data augmentations would be used. Data augmentations are techniques to slightly change the image by rotating, zooming, changing the lighting, etc. to artificially create a new image. These techniques are known to make the model more robust as it sees multiple slightly different versions of the same image. Here we went with the standard data augmentations as defined in the fast.ai library.

ITERATIONS IN THE MODELLING PROCESS

The first iteration was in the labelling of the images. In the end, we want to predict contaminated vs. non-contaminated. However, as explained above, there are several "anomalies" that could happen, which all had to be classified either in

contaminated or non-contaminated. Therefore, we tried to turn the problem in a multi-class problem by making one class per anomaly, i.e. we had 8 classes: No_CFU, CFU, Corrupted_Agar, Air_Bubble, Tape, Marker, Glue, and Tag. Indeed, it might would have been easier for a model to separate 8 homogeneous classes, rather than 2 heterogeneous classes. However, after some iterations, we saw that the model obtained better results when only 2 classes (Contaminated vs Non-Contaminated) were used, rather than 8.

Next, we iterated on different hyper-parameters for the model itself. Multiple loss functions (a loss function is a function that is used to evaluate if the model is heading in the right direction (i.e. making correct predictions) and penalizing the model if it is not heading in the right direction) were tested and the best results were obtained with the "standard" cross-entropy loss with class weights and with the Focal loss function [22]. Actually, the focus of this project was on reducing as much as possible the number of false negatives (i.e. images predicted as being non-contaminated, but actually are contaminated). Therefore, it was important to investigate if different loss functions would result in this desired behaviour.

Thirdly, we investigated what network architecture and size (i.e. number of layers in the network) would give the best results. The bigger the network, the more fine-grained and complex patterns the network is able to detect, but also the more time it takes to train the network. Also, different network architectures, will bring different results. Here we tested the major networks including: ResNet, Inception, VGG, DenseNet. We obtained the best results with the ResNet architecture, followed closely by the Inception architecture. Finally, we observed that the bigger networks (ResNet152) had better results than the smaller ones (ResNet30, ResNet50, ResNet101).

Fourthly, we found out that we were having better results if we made a separate model for the petri dishes of 55 mm and the ones of 90 mm. This makes sense, as for the images of 55 mm you have a smaller petri dish and more "unnecessary" information on the image, i.e. the border parts of the image where no petri dish is located, as can be seen in Figures 2 and 3.

Figure 2: a 55-mm petri dish with a CFU

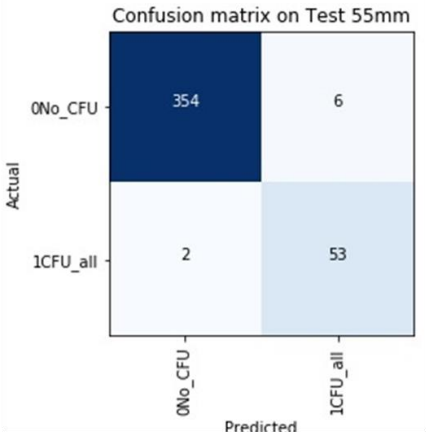
Figure 3: a 90-mm petri dish with a slightly corrupted agar

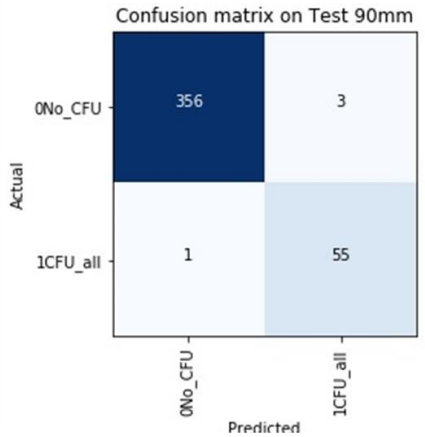
Finally, having still too much false negatives, we started to investigate what were the cases in which the model had difficulties to predict the right class. These were the cases of slightly corrupted agar and very small CFUs or CFUs only on the border of the petri dish. Consequently, we created some extra instances of these unusual cases, in order to over-represent or over-sample them. This in the hope that the model would get better by getting to see more examples. This, indeed, helped the model to obtain better performances and obtain less false negatives. However, we also introduced some bias in the model without knowing it. Actually, the images of the unusual cases were taken at a later date than all the others and there was something, which isn't visible from the human eye, that changed at that particular date in the images. Moreover, as we only fed new positive (contaminated) images to the model of that particular date, the model assumed that this change in the images could always be considered as contaminated petri dishes, as it did not see this change in non-contaminated images. Hence, all new images we made to test the model, were all predicted as being contaminated, just because they all were created after that particular date. This is one of the dangers of deep-learning models, i.e. they take decisions based on patterns not always visible to the human eye. Which makes them so powerful, but also very dangerous if they are not tested properly. Therefore, it is important to design sound training, testing and validation sets. Thus, we reshuffled all the images in the different training, validation, and testing sets, by making sure that there were both positive (contaminated) and negative (non-contaminated) examples of each date in

the different sets, to make clear to the model, that differences in the capturing software, are not a sign of contamination or not of the petri dish. We then obtained better results than initially and had eliminated the bias in the model.

THE RESULTS

We obtained very good results with the two models we trained (one for each size of petri dish) on our test data as can be seen below on Figure 4 below. We used a worst-case scenario, by taking inside the test set some of the most difficult images (CFU on the border) and over-representing them, to test the model at its limit. In the real production case, those difficult cases rarely occur (there was no occurrence of a CFU on the border without any other CFU on the rest of the petri dish, in the last 8 months), however they could appear. Therefore, these results have to be taken with a pinch of salt, as we expect much better results in real production situation. You can see on top of the Figure 4, two confusion matrixes. These should be interpreted in the following way: horizontally you can see the actual or true label (0No_CFU = non-contaminated; 1CFU_all = contaminated) and vertically you can see the predicted label. For instance, on the confusion matrix for the 55-mm petri dishes, the number 354 refers to the true negatives, i.e. 354 images were predicted as non-contaminated and were non-contaminated. The 6 refers to the false positives, i.e. 6 images were predicted as being contaminated but were not. The 2 represents the false negatives, i.e. 2 images were predicted as being non-contaminated but were. And finally, the 53 are the true positives, i.e. 53 images were predicted as being contaminated and were contaminated.





Size	CrossEntropy Loss	Accuracy	Cohen Kappa	False Negatives
55 mm	0.0741	98.07%	0.9187	3.64%
90 mm	0.1529	99.04%	0.9593	1.79%

Figure 4: Results on the test set using a worst-case scenario

Let's go over the different metrics reported in Figure 4. The Cross-Entropy Loss is a loss measure used in deep learning models that is meant to serve as objective function to minimize. Next, the Accuracy is a measure that counts how much percent of the time you predict something correctly. Then, the Cohen Kappa [23] is a measure that considers class-imbalances. Actually, if you have a case where in the sample you have 99% of the examples are negative and only 1% is positive, you can have a classifier that always predicts negative and that would result into an accuracy of 99%. However, you would do a really bad job, as you would identify none of the positives. This is a case of class imbalance (there are much more negatives than positives). In these cases, you need to find other metrics that are more appropriate and take these class imbalances into account, such as the Cohen Kappa. The Cohen Kappa ranges from -1 to +1, with 0 being random performance, +1: perfect performance, positive: better than random and negative: worse than random performance. Finally, the False Negative rate is calculated by taking all the false negatives (i.e. predicted as being non-contaminated, but was contaminated) and dividing them by all the positives (i.e. contaminated).

Even though, the model still makes some mistakes, we believe that those mistakes are minor and are less than the mistakes a human operator would make. Imagine yourself having to check thousands of petri dishes each day, you can imagine that you would miss some of the cases. Currently, this has not been tested yet, but it is the goal to test the performances of humans vs. the machine in a future phase of the project.

Finally, we needed to make sure our solution would fit in the current architecture and environment, from a technical, legal and user's point of view. This was done by organising multiple workshops with the different stakeholders. In these workshops, we would brainstorm and discuss several ideas on how this integration would be the most efficient possible.

MODEL VISUALISATIONS

In addition to these results, we developed visualisations on the images to show and try to explain what the model sees and on what it bases its decisions, as can be seen in Figure 5 below. The first visualisations we made are called Gradient-weighted Class Activation Mapping (Grad-CAM) [24]. These are heatmaps on top of the image, that highlight which pixels were the most important in making the prediction. Consequently, you can interpret this as, these highlighted pixels have led us to make this decision. This has a double utility. Firstly, this helps us to explain to non-technical users how the model behaves and what parts of the image it uses to take its decisions. Secondly, it also helps us (data scientists) to check if the model is not biased and uses wrong parts of the image to make its decision. For instance, in the case described above, when we introduced bias by adding only positive images after a particular date in the training set, we could see that the model was basing its decision to always predict positive, with always the same particular area in the image. Consequently, this helped us to understand that something had changed in that specific area in the image, which introduced the bias in the model.

Furthermore, we created a second visualisation based on convolutions. As explained before, a convolution is basically a matrix multiplication between a square of pixels on the image and a filter (i.e. a weight matrix). This is the essential building block of a CNN. We chose a particular filter, that has the property to show the edges in the image. This is not necessarily a filter that is used in our model, however, for humans it is a handy filter to see what the model can potentially use to make its decisions.

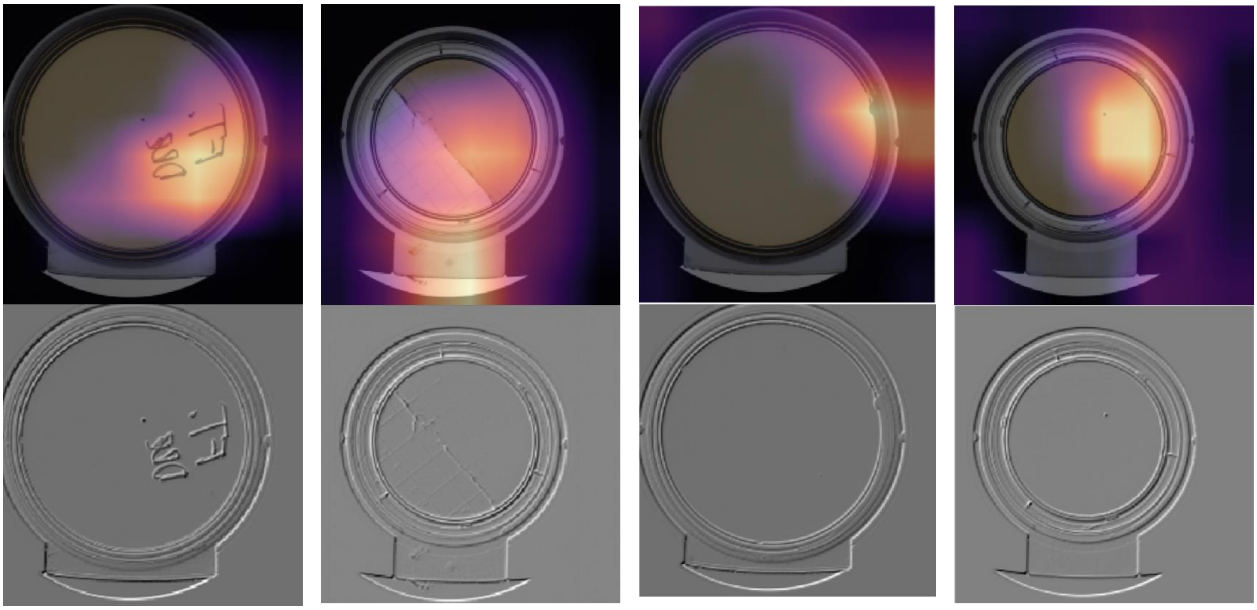


Figure 5: Model visualisations. In the top row, you can see the Grad-CAM visualisations and in the bottom row, the Convolution ones, for 4 different petri dishes. From left to right, a 90-mm petri dish with marker annotations, a 55-mm petri dish with corrupted agar, a 90-mm petri dish with a CFU on the top right border and a 55-mm petri dish with a small CFU in the upper right part.

CONCLUSION

The question is not anymore: “Is the deep learning technology ready?”, because it is, but the question is more: “Are you ready to embrace the disruptive advances in the deep learning technology?”. We have demonstrated in this paper that by using the right techniques with skilled people, you can arrive to first-class results. However, this is not a road without any hurdles, but these hurdles can be overcome. We saw that by using transfer learning, you win a huge step in performances, as you do not have to start from scratch. Without any transfer learning, you would start at an accuracy of 5 to 10% compared to 80 to 90% with transfer learning. In addition, one must keep an eye open for bias in the model, by designing sound training, validation and testing set and by checking through visualisations (such as Grad-CAMs) the validity of the model decisions. Moreover, using an iterative (agile) process in combination with continuous collaboration with the business stakeholders can avoid a misfit of the model to the business need and ensures a smooth, but steady progress. Finally, you need to make sure you can integrate your solution in the actual setting. This was done by organising multiple workshops with the different stakeholders. Consequently, we consider the deep learning

approach as an elevated value added to multiple company processes, with high potentials of efficiency increases and monetary savings.

REFERENCES

[1] Ferrari A., Lombardi S., Signoroni A., Bacterial colony counting with Convolutional Neural Networks in Digital Microbiology Imaging, Pattern Recognition, Vol. 61, January 2017, Pages 629-640

[2] Schmidhuber, J., Deep learning in neural networks: An overview, Neural Networks, 2015, Vol. 61, Pages 85-117

[3] Chui M, Kamalnath V. McCarthy B., An executive's guide to AI, McKinsey & Company, 2019

[4] S. Novak, E. Marlowe, Automation in the clinical microbiology laboratory, Clin. Lab. Med. 33 (3) (2013) 567–588.

[5] C.D. Doern,M. Holfelder, Automation and design of the clinical microbiology laboratory, in: Manual of Clinical Microbiology, 11th edition, ASMPress, Washington, DC, 2015, pp.44–53.

[6] A. Ferrari,S. Lombardi,A. Signoroni,Bacterial colony counting by convolutional neural networks, in: 37th Annual international Conference of the IEEE Engineering in Medicine and Biology Society EMBC 2015, 2015, pp.7458–7461.

[7] S. Brugger, C. Baumberger, M. Jost, W. Jenni, U. Brugger, K. Mühlemann, Automated counting of bacterial colony forming units on agar plates, PLoS One 7(3), 2012.

[8] C. Zhang, W. Chen, W. Liu,C. Chen, Anautomated bacterial colony counting system, in: IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC2008),11–13 June 2008, Taichung, Taiwan, 2008, pp.233–240.

[9] H.P. Mansberg, Automatic particle and bacterial colony counter, Science 126 (3278) (1957) 823–827.

[10] H.E. Kunitschek, Electronic counting and sizing of bacteria, Nature 182(4630) (1958)234–235.

[11] N. Alexander, D. Glick, Automatic counting of bacterial cultures - a new machine, IRETrans. Med. Electron. PGME-12(1958)89–92.

[12] D. Mukherjee, Bacterial colony counting using distance transform, Int. J. Biomed. Comput. 38(1995)131–140.

[13] G. Corkidi, R. Diaz-Uribe, J. Folch-Mallol, J. Nieto-Sotelo, Covasiam: an image analysis method that allows detection of confluent microbial colonies and colonies of various sizes for automated counting, Appl. Environ. Microbiol. 64 (4) (1998)1400–1404.

[14] A. Liu, Z. Liu, L. Song, D. Han, Adaptive ideal image reconstruction for bacteria colony detection, in: E. Zhu, S. Sambath (Eds.), Information Technology and Agricultural Engineering, Advances in Intelligent and Soft Computing, vol.134, Springer, Berlin, Heidelberg, 2012, pp.353–360.

[15] A.B.G.L. Masala, U. Bottigli, Automatic cell colony counting by region-growing approach, Il Nuovo Cimento C(2008) 633–644.

[16] Q. Geissmann, Opencfu, a new free and open-source software to count cell colonies and other circular objects, PLoS One 8 (2) (2013) e54072.

[17] A. Ferrari, A. Signoroni, Multistage classification for bacterial colonies recognition on solid agar images, in: 2014 IEEE International Conference on Imaging Systems and Techniques (IST), 2014, pp.101–106.

[18] Balázs Csanád Csáji, Approximation with Artificial Neural Networks; Faculty of Sciences; Eötvös Loránd University, Hungary, 2001

[19] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L., ImageNet: A large-scale hierarchical image database. In CVPR, 2009.

[20] Shearer C., The CRISP-DM model: the new blueprint for data mining, J Data Warehousing, 2000; 5:13—22.

[21] Cody A. Coleman, Deepak Narayanan, Daniel Kang, Tian Zhao, Jian Zhang, Luigi Nardi, Peter Bailis, Kunle Olukotun, Chris Ré, and Matei Zaharia, DAWNBench: An End-to-End Deep Learning Benchmark and Competition, NIPS ML Systems Workshop, 2017

[22] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, Piotr Dollár, Focal Loss for Dense Object Detection, IEEE International Conference on Computer Vision (ICCV), 2017

[23] Pontius, Robert; Millones, Marco, "Death to Kappa: birth of quantity disagreement and allocation disagreement for accuracy assessment". International Journal of Remote Sensing. 32 (15): 4407–4429, 2011.

[24] Ramprasaath R. Selvaraju, Michael Cogswell; Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization, IEEE International Conference on Computer Vision (ICCV), 2016

ACKNOWLEDGMENTS

I would like to thank all the consultants and sales at Business & Decision involved in this project. It was a very interesting and challenging road we undertook, but with very promising results! I would also like to thank our client, with whom we had a very fruitful collaboration, with great learning-outcomes for both parties.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Thomas Dekelver
Business & Decision Life Sciences
Sint-Lambertusstraat 141
Brussels / 1200
Email: Thomas.dekelver@businessdecision.be
Web: https://www.businessdecision-lifesciences.com/