Thesis for the Degree of Doctor of Philosophy

Unsupervised and Semi-supervised Learning Methods based on Deep Clustering and Explainable Active Learning

by

Caleb Vununu

Department of IT Convergence and Application Engineering

The Graduate School

Pukyong National University

August 2021

Unsupervised and Semi-Supervised Learning Methods based on Deep Clustering and Explainable Artificial Intelligence
딥 클러스터링 및 설명 가능한 능동 학습을 기반으로 한 비지도 및 반지도 학습 방법

Advisor: Prof. Ki-Ryong Kwon

by

Caleb Vununu

A thesis submitted in partial fulfillment of the requirements for the degree
of

Doctor of Philosophy

in Department of IT Convergence and Application Engineering, The Graduate School, Pukyong National University

August 2021

Unsupervised and Semi-supervised Learning Methods based on Deep Clustering and Explainable Active Learning

A dissertation

by

Caleb Vununu

Approved by:

(Chairman) Prof. Heung-Kook Choi

(Member) Prof. Bong-Kee Sin	(Member) Prof. Oh-Heum Kwon
(Member) Prof. Suk-Hwan Lee	(Member) Prof. Ki-Ryong Kwon

August 27, 2021

Contents

I. INTRODUCTION 1

1.1 Research motivation 1

1.2 Thesis plan 4

II. LITERATURE REVIEW 6

2.1 Unsupervised learning: General view 6

2.2 Deep clustering 10

2.3 Explainable artificial intelligence 12

2.3.1 Overview 12

2.3.2 Relevance maps 14

2.3.3 Evaluation of relevance maps 17

2.4 Active learning 19

2.5 Unsupervised learning in medical images analysis 23

2.6 HEp-2 cell classification in literature 24

III. PROPOSED METHODS 28

3.1 Proposed deep clustering method 28

3.1.1 Method overview 28

3.1.2 Deep embedded clustering network 29

3.1.3 Deep embedded clustering with dual autoencoder 38

3.1.4 Attention-based deep clustering 41

3.2 Proposed active learning method 45

3.2.1 Method overview 45

3.2.2 Deep parallel residual networks 45

3.2.3 Active learning using the pre-trained parallel residual networks 54

3.2.4 Explainable active learning 57

IV. EXPERIMENTAL RESULTS AND DISCUSSION 64

4.1 Deep clustering results 64

4.1.1 Datasets and experimental setups 64

4.1.2 Results for Case-1 68

4.1.3 Results for Case-2 70

4.1.4 Results for Case-3 72

4.2 Active learning results 82

4.2.1 Datasets and initial setups 82

4.2.2 Results of the "RS" case 88

4.2.3 Results of the "AL" case 91

4.2.4 Results of the "IN-RS" case 93

4.2.5 Results of the "IN-AL" case 95

4.2.6 Results of the explainable active learning 102

V. CONCLUSIONS 119

VI. REFERENCES 122

VII. ACKNOWLEDGEMENT 137

Figures

Fig. 1. Data importance in deep learning. 1

Fig. 2. Pattern recognition framework. 7

Fig. 3. Illustration of the denoising autoencoder. 8

Fig. 4. Illustration of the deep convolutional autoencoder (DACE). 9

Fig. 5. Schematic representation of XAI methods. 13

Fig. 6. Examples of relevance maps produced by LRP: 15

Fig. 7. Examples of relevance maps produced by DeepTaylor 17

Fig. 8. Schematic representation of the conventional active learning method.22

Fig. 9. Illustration of the proposed method. 29

Fig. 10. Illustration of the pooling indices storage technique. 35

Fig. 11. The connections between the pooling layers from the encoder and the unpooling layers from the decoder. 36

Fig. 12. Illustration of the proposed network using the storage and copy\concatenate techniques at the same time 37

Fig. 13. Illustration of the proposed dual embedded network 39

Fig. 14. Architecture of the attention-based network. 42

Fig. 15. Illustration of the proposed deep attention-based network. 43

Fig. 16. HEp-2 cellular images from the 13A dataset. 46

Fig. 17. Illustration of the proposed parallel networks 47

Fig. 18. Approximation coefficients of the HEp-2 cellular images from the SNPHEp-2 dataset 48

Fig. 19. Details coefficients of the HEp-2 cellular images from the SNPHEp-2 dataset..49

Fig. 20. Architecture of the residual networks...51

Fig. 21. Illustration of fine-tuning 53

Fig. 22. Illustration of the different steps of the proposed active learning scheme 56

Fig. 23. Illustration of the different steps of the proposed explainable active learning...59

Fig. 24. Example images from the SNPHEp-2 dataset..................................65

Fig. 25. Visualization of the features learned by the DCAE of case-1............68

Fig. 26. Confusion matrix for case-1 69

Fig. 27. Visualization of the features learned by the DCAE of case-2............70

Fig. 28. Confusion matrix for case-2 71

Fig. 29. Visualization of the features learned by the DCAE of case-3............73

Fig. 30. Confusion matrix for case-3 73

Fig. 31. Clustering accuracy of the three networks.......................................74

Fig. 32. The global loss of the networks...75

Fig. 33. Accuracy of the three networks...77

Fig. 34. Example images from the large-scale HEp-2 dataset 83

Fig. 35. Results of the initial learning process 85

Fig. 36. Classification results of the "RS" case...89

Fig. 37. Classification results of the "AL" case 92

Fig. 38. Classification results of the "IN-RS" case 94

Fig. 39. Classification results of the “IN-AL” case 96
Fig. 40. Classification accuracy of the three most difficult cellular patterns...97
Fig. 41. Evolution of the accuracy with other limitations in terms of labeling 98
Fig. 42. Accuracy improvement with the initial learning in case of 100% of training data available 99
Fig. 43. Evaluation of the different relevance maps methods.......................103
Fig. 44. Results on the LSHEp-2 dataset 105
Fig. 45. Comparison between the random sampling (RS), conventional active learning (CAL) and the proposed explainable active learning (XAL) ……………………………………………………………………..105
Fig. 46. Homogeneous images with their corresponding maps: 106
Fig. 47. Example of some homogeneous images, with their corresponding maps, that were selected by the XAL..107
Fig. 48. Speckled images with their corresponding maps: 108
Fig. 49. Example of some speckled images, with their corresponding maps, that were selected by the XAL. 109
Fig. 50. Nucleolar images with their corresponding maps: 110
Fig. 51. Example of some nucleolar images, with their corresponding maps, that were selected by the XAL. 111
Fig. 52. Centromere images with their corresponding maps: 112
Fig. 53. Example of some centromere images, with their corresponding maps, that were selected by the XAL..113
Fig. 54. Nuclear membrane images with their corresponding maps: 114

Fig. 55. Example of some nuclear membrane images, with their corresponding maps, that were selected by the XAL..115

Fig. 56. Golgi images with their corresponding maps:.................................116

Fig. 57. Example of some Golgi images, with their corresponding maps, that were selected by the XAL. 116

Fig. 58. Comparison using the CIFRA-10 dataset 117

Tables

Table 1. Architecture of the DCAE. 33

Table 2. The different steps of selection using the conventional active learning scheme. 55

Table 3. The different steps of the proposed explainable active learning. 600

Table 4. Comparative study for the SNPHEp-2 dataset. 79

Table 5. Comparative study for the 13A dataset. 800

Table 6. The different cases used during the experiments. 877

Table 7. Comparative study using the LSHEp-2 dataset. 1000

딥 클러스터링 및 설명 가능한 능동 학습을 기반으로 한 비지도 및 반지도 학습 방법

Caleb Vununu

부 경 대 학 교 대 학 원 IT 융 합 응 용 공 학 과

요 약

최근 다양한 영역에서 성능 측면의 딥 러닝 연구가 활발히 진행되면서 광범위한 빅데이터의 사용이 가능해졌다. 점점 더 많은 산업 또는 의료 분야에서 지능형 시스템들은 수천~수만 개의 데이터로부터 학습된 기계 학습 또는 딥 러닝 알고리즘으로 추출된 정보를 기반으로 한다. 특히 딥 러닝 기반 방법은 정확성과 효율성을 위해 지나치게 많은 양의 데이터를 필요하므로, 지능형 시스템을 구축하는 데 필요한 데이터의 수는 계속해서 기하 급수적으로 증가할 것이다. 이와 같은 상황과 더불어 데이터 라벨링의 문제가 제기된다. 대부분의 딥 러닝 응용 프로그램, 특히 의료 분야에서는 지도 학습(supervised learning)을 기반으로 한다. 지도 학습은 매우 우수한 식별력의 결과에 도달하지만, 수천~수만 개의 레이블이 지정된 데이터들을 보유해야 한다. 그러므로 기하 급수적으로 증가되는 과업의 학습 시스템에서 데이터 라벨링의 문제점들이 나타나게 된다.

지도 학습의 높은 식별 성능을 유지하면서, 데이터 라벨링 과정이 필요없거나 감소시킬 수 있는 비지도 학습과 반지도 학습에 이르기까지 다양한 학습 방법에 대한 연구가 필요하다. 본 연구에서는 HEp-2(Human Epithelial type-2)의 인간 상피 유형 세포에 대한 분류 과업에 중점을 두고, 비지도 학습과 반지도 학습에 대한 서로 다른 시나리오를 기반으로 인공지능 학습 모델을 제안한다.

첫 번째 제안 모델은 딥 클러스터링(deep clustering) 기반 지도 학습 모델로, 모델의 종단(end-to-end) 간 학습을 수행하기 위해 라벨링 데이터가 전혀 필요하지 않

는 완전 비지도 학습(fully unsupervised learning) 방법이다. 딥 클러스터링은 기존 클러스터링 방법과 딥 러닝 구조를 결합한 것으로, DCAE(Deep Convolutional Autoencoder)의 중간에 잠재 공간(latent space)의 특징들을 모든 학습 과정에서 클러스터링하는 계층을 결합한 것이다. 제안한 딥 클러스터링은 다음과 같은 주요 특징을 가진다. (1) DCAE가 쉽게 식별할 수 있는 잠재 특징들을 생성할 수 있도록 한다. (2) DCAE의 복원이 극대화되지 않는 기존 SOTA(state-of-the-art)의 딥 클러스터링 방법과는 달리, DCAE의 복원이 최대화되는 어텐션(attention) 기반 딥 클러스터링 아키텍처로 설계된다. (3) 딥 클러스터링의 잠재 특징들에 대한 복원 정확도 영향을 분석한다.

두 번째 제안 모델은 데이터 라벨링 과정을 완화하면서, 설명이 가능한 능동 학습(explainable active learning) 기반의 준지도 학습 방법이다. 능동 학습은 사용 가능한 모든 데이터에 레이블을 지정할 필요없이, 전체 모델의 정확도를 최대화하는데 도움이 되는 제한된 수의 데이터를 선택하는 것으로 구성된다. 제안한 방법에서는 라벨링이 필요한 데이터를 선택하기 위하여 사전 선택된 Explainable AI (XAI) 방법에 의해 생성된 관련성 맵(Relevance Map)이 사용하여, 능동 학습을 재정의한다. 제안한 능동 학습 방법은 다음과 같은 주요 특징을 가진다. 1) 데이터 선택 과정이 기존 능동 학습 방법에서와 같이 모델의 정확도에 의존하지 않기 때문에 성능에 무관하게(performance-agnostic) 과업을 수행할 수 있다. 2) XAI를 사용함으로써, 모델의 전체 평가에 기여할 수 있는 데이터 선택 과정의 설명 가능성을 보장한다.

제안한 딥 클러스터링 기반의 비지도 학습 방법과 설명 가능한 능동 학습의 반지도 학습 방법은 기존 HEp-2 세포의 공개 데이터 셋에 대하여 광범위하게 테스트되었다. 이들 실험 결과로부터 본 연구의 주요 공헌은 다음과 같다.

제안한 딥 클러스터링 모델의 경우 (1) DCAE의 복원을 강화하기 위해 풀링색인 저장(pooling indices storage), 복사와 연결(copy and concatenation), 어텐션 기반 네트워크(attention based network) 등의 다양한 학습 기술들을 적용하며, 딥 클러스터링 SOTA 방법보다 우수한 복원 정확도를 가짐을 확인하였다. (2) 실험으로부터 DCAE의 복원 성능이 클러스터링 정확도에 대한 영향을 분석하였다. 분석으로부터, DCAE의 우수한 복원 성능이 학습된 잠재 특징들의 품질 향상됨을 확인하였다. 제

안한 어텐션 기반 네트워크 모델이 기존의 handcrafted features 보다 우수한 품질을 가지며, HEp-2 세포 분류 과업에서 지도 학습 방법과 유사한 수준인 데이터 셋 중 하나에서 97.56 %까지 도달함을 확인하였다.

제안한 설명 가능한 능동 학습의 경우 (1) 기존 방법에서 모델의 출력을 사용하는 대신 XAI 기반 관련성 맵 기반의 데이터 선택 방법을 제안함으로써 설명 가능성의 잇점을 제공한다. 이는 딥 러닝 모델이 제기하는 해석성 문제의 맥락에 매우 중요한 성질이다. (2) XAI 기반 데이터 선택 방법이 매우 적은 수의 라벨링된 데이터로부터 모델의 성능을 향상시키는 데 기여한다. 실험으로부터 20%에 불과한 훈련 데이터만으로부터 제안한 방법이 대규모 HEp-2 세포 데이터 셋에서 92.77 %의 정확도를 가졌으나, 랜덤 선택 방법 및 기존 선택 방법들은 각각 82 %와 90 %의 정확도를 가짐을 확인하였다.

Unsupervised and Semi-supervised Learning Methods based on Deep Clustering and Explainable Active Learning

Caleb Vununu

Department of IT Convergence and Application Engineering,
The Graduate School,
Pukyong National University

Abstract

In the recent years, the progress of deep learning in terms of performance in various areas has enabled the extensive use of big data. More and more intelligent systems, either in industrial or medical areas, are based on the information extracted by machine learning algorithms from thousands of data. As deep learning based methods require an excessively large amount of data in order to be efficient, we can expect that the number of data needed in order to build intelligent systems will only continue to increase exponentially. This situation poses a challenge in terms of data labeling. In fact, most of the deep learning applications, especially in the medical area, are based on the supervised learning. Even though the supervised learning paradigm reaches tremendous results in terms of discrimination, the necessity of having thousands of labeled images represents a serious drawback for this kind of learning system.

The aim of the present study is to explore different methods, from deep clustering to semi-supervised learning techniques, that can completely eliminate or substantially diminish the need of data labeling process while maintaining a high performance in terms of discrimination. Our study focuses on the classification of the different types of the Human Epithelial type-2 (HEp-2) cells. We propose two models that are based on different scenarios.

The first model is based on an unsupervised learning paradigm (scheme). We can say that the method is fully unsupervised in the sense that there is no need of labeled data in order to perform an end-to-end training of the model. On that purpose, we use deep clustering, a technique that combines conventional clustering methods with deep learning structures. A clustering layer is incorporated in the middle of a deep convolutional autoencoder (DCAE) in order to perform clustering on the latent space's features at every iteration of the training process. This technique allows the DCAE to produce latent features that are easily discriminable. Furtherly, unlike in the actual state-of-the-art deep clustering methods, where the reconstruction of the DCAE is not maximized, we propose a novel architecture for deep clustering, named attention-based deep clustering, where the DACE's reconstruction accuracy is efficiently maximized. Finally, for this first model, we present a systematic analysis of the results in order to explore the influence of the reconstruction accuracy on the latent features.

The second model is based on a semi-supervised learning scheme. We adopt the techniques of active learning in order to alleviate the data labeling process. Active learning consists of applying some methods in order to select, among the thousands of available data, only those that really need to be labeled. It consists of selecting a limited number of informative data that can help to maximize the overall model's accuracy without the exigency of labeling the totality of the available data. In this work, we redefine active learning by proposing a data selection process based on explainable artificial intelligence (XAI). Relevance maps produced by some pre-selected XAI methods are used in order to find the informative data that need to be labeled. This proposed active learning scheme is performance-agnostic, since the data selection process does not depend on the accuracy of the model like in the conventional active learning methods. Secondly, by using XAI, we ensure the *explainability* of the selection process, which can contribute to the overall understating of the model. The two proposed models, the deep clustering and the explainable active learning, are extensively tested on the existing HEp-2 cell public datasets.

The key contributions of the present work can be summarized as follows. For the deep clustering model, (1) we propose different techniques (pooling indices storage, copy and concatenation and attention network) in order to boost the reconstruction of the DCAE. Our method outperforms the state-of-the-art deep clustering method in terms of reconstruction accuracy. (2) We investigate if the reconstruction quality can affect the clustering accuracy. (3) We demonstrate that a better reconstruction of the DCAE increases the quality of the learned latent features. Our best model (attention network) outperforms the conventional handcrafted features and reaches similar level (97.56% on one of the datasets) with the supervised learning methods in terms of discrimination of the HEp-2 cells.

For the proposed explainable active learning, (1) we propose a new definition of the selection method. Instead of using the model's output, as normally done in the conventional methods, we propose the use of the relevance maps. (2) Following the obtained results during the experiments, this selection method provides better results and, more importantly, has the advantage of the explainability. This is very important in the context of interpretability issues posed by the deep learning models. (3) We demonstrate that, using our proposed active learning method, an XAI-based selection contributes to boost the performance of the model even with a quite limited number of labeled data. With only 20% of the training data, the discrimination results achieve 92.77% of accuracy in one of the large HEp-2 cell dataset, while random and conventional selections give, respectively, 82% and 90%.

I. INTRODUCTION

1.1 Research motivation

In the past decade, deep learning has proven to be very efficient in different kinds of applications. Since at least the beginning of the past decade, many researchers in various areas have adopted deep learning in order to tackle many different kinds of problems. Problems in computer vision include analysis of medical images also. We know that deep learning-based models require thousands of images.

Since 2014-2015 and the "revolution of depth", it has been proven that the performance of models using deep learning can increase with the availability of the data. Which means that, not only you need a quite huge amount of data in order to construct a deep learning-based model in the first place but, also, the performance of your model can increase if you use more and more data. As we can see in Fig. 1, compared to other machine learning techniques, the performance of deep learning-based methods can really reach tremendous limits in case you use many data.

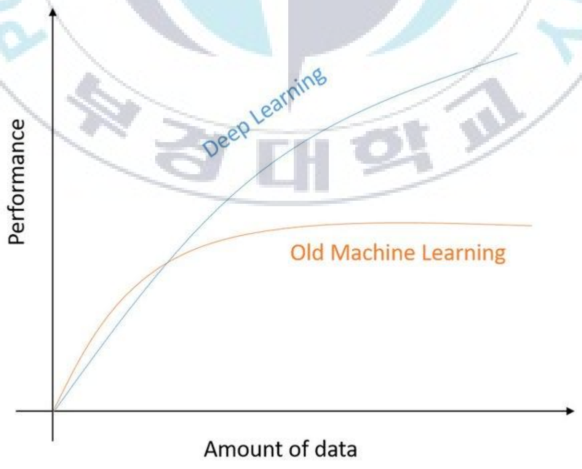


Fig. 1. Data importance in deep learning.

In case of the diagnosis of autoimmune diseases, for example, one of the most important steps in the computer-aided diagnosis (CAD) systems is the automatic classification of the images representing the different human epithelial of type 2 (HEp-

2) cell types [1]. During the past decade, researchers have proposed numerous different methods for the automatic classification of HEp-2 cellular images. As a normal pattern recognition problem [2], HEp-2 cell image classification methods comprise mainly two distinctive steps: feature extraction and discrimination (referred also as classification). Moreover, two different approaches can be separated in the literature: conventional machine learning methods based on handcrafted features and the deep learning methods based on the automatic feature learning approach.

Methods based on the conventional machine learning scheme typically present a certain feature extraction approach followed by a certain classification model. The feature extraction part utilizes some specific information supposed to contain the inherent characteristics of the data. Conventional handcrafted features cover different areas of computer vision pre-processing techniques. And, finally, the performance of these approaches specifically depends on the discrimination potential carried by the chosen features.

The second group of methods concerns the deep learning-based ones. The first advantage of the deep learning-based features is to provide an automatic way of feature learning that does not depend on the subjective choice of the researcher. Deep learning has proven to be effective in many different areas. And, many researchers in the CAD literature have adopted the deep learning paradigm over the past decade. Especially in the HEp-2 cells classification literature, handcrafted features were completely supplanted by the deep learning-based ones.

The most important point to note about these deep learning-based methods is that they all utilize the supervised learning paradigm. The supervised learning approach has the exigency of having images labeled manually by the experts in order to train the network. This can represent a drawback for these methods, knowing that deep learning structures necessitate thousands of images. Labeling manually this huge quantity of images can represent a quite challenging and burdensome task. On the other hand, the unsupervised learning paradigm presents the advantage of performing the feature extraction without the need of any labeled data during the training.

In our previous work [3], we investigated an unsupervised learning scheme for the HEp-2 cell classification. However, only the feature extraction part was completely undertaken in an unsupervised learning way. The created features were given to a nonlinear classifier that learned with the use of the manually labeled data. Thus, while the feature extraction was undertaken in an unsupervised way, discrimination was still required to be supervised. The first contribution of the present work is to propose a scheme where the feature extraction and the discrimination parts are both performed in a strictly unsupervised paradigm.

We adopt the deep convolutional autoencoder (DCAE) as the principal feature extractor. The DCAE takes the original images as inputs and learns how to reproduce them via an encoding–decoding structure. Unlike in our previous work, where the features learned by the DCAE were extracted and then utilized as the inputs of a nonlinear classifier, we propose to embed a clustering layer in the DCAE. The clustering layer will learn, in every single step of the training process, how to automatically discriminate the latent representations produced by the DCAE.

The principal aim of this work is to present discrimination methods for the HEp-2 cell images that minimize the need of the labels. For that purpose, we first propose an unsupervised learning method that uses deep clustering and, secondly, a semi-supervised learning method that uses explainable artificial intelligence (XAI). In terms of contributions, we present a deep clustering method that focuses on the reconstruction quality and we present a quite detailed investigation about the effects of the reconstruction quality on the clustering performance. We investigate how the autoencoders' reconstruction process can affect the clustering process.

In terms of contributions for the second method, we present an active learning technique that completely abandons the use of the model's output and we try to formulate a new selection process that can be explainable. In fact, we use the relevance maps (also referred as relevance maps or attribution maps) generated by the XAI methods and try to utilize them for the selection process. Detailed experiments are conducted on the HEp-2 cell public datasets in order to evaluate the effectiveness of the proposed original selection method.

1.2 Thesis plan

In chapter 2, the literature review part, we discuss in details the conventional and state-of-the-art methods for the two subjects. Firstly, in section 2.1, we present a general review of different conventional unsupervised learning methods. Then, in section 2.2, we introduce the concept of deep clustering. A general review is made about deep clustering by analyzing the differences between “conventional clustering” and “deep clustering” methods. Note that we also highlight the existing differences between deep learning-based clustering and deep clustering.

In section 2.3.1, we introduce and discuss in general the basic concepts of explainable artificial intelligence (XAI). In section 2.3.2, we introduce the concept of relevance maps and discuss about the different methods than can help to generate them. In section 2.3.3, we present succinctly different methods that can help to evaluate these relevance maps. In section 2.4, we introduce the general concept of active learning and also we present a brief review of the conventional active learning methods.

In section 2.5, our aim was to give a brief overview of the different applications of unsupervised learning methods in the area of medical image analysis. Since our experiments concentrate on these kinds of images, especially the HEp-2 cell images, we want to give a perspective of how unsupervised learning methods are utilized nowadays in this area. In section 2.6, we go through the detailed literature review of the different methods presented for the classification of the HEp-2 cells.

In fact, in the HEp-2 cell classification literature, the quasi-majority of the methods prefer to tackle this problem by using the supervised learning way. Our goal is to prove that unsupervised learning methods in this area, if properly designed, can achieve at least the same performance as the actual state-of-the-art supervised learning methods.

The third chapter will present the proposed methods. The first part (section 3.1) will present in details each step of the formulation of the proposed deep clustering method. We present also different types of networks that can be used with our loss functions. In the second part (section 3.2), we present our proposed active learning method.

The fourth chapter presents the experimental results. Here also, we divided the chapter into two sections. Section 4.1 presents in details the results of the proposed deep clustering method. We present the results step by step for the different cases discussed in chapter 3. We also conduct in the same time an exhaustive comparative study with the other methods in the literature.

In section 4.2, we present the results obtained by applying our proposed active learning method. Here also, we present the results by steps, by showing how the performance can increase while adopting the proposed selection method.

Because the proposed active learning method involves explainable artificial intelligence, we first show briefly how we selected the relevance maps' methods. Some experiments are conducted before the active learning scheme precisely in order to select the best XAI methods that produce relevance maps that are really relevant for our data and trained models.

II. LITERATURE REVIEW

2.1 Unsupervised learning: General view

In machine learning, unsupervised learning regroups all the methods and techniques that learn the inner characteristics of the data without any supervision. In case of discrimination tasks, the supervision comes from the use of labels. On the contrary, the absence of labels obligates the learning method to explore and reveal, by itself, all the elements that define the essence of the original data. In this brief review, we separate the unsupervised learning methods into two main groups: methods that perform cluster analysis and the ones that perform dimensionality reduction.

Cluster analysis (or just clustering) is the unsupervised version of classification, which means that clustering is the process of performing discrimination without the help of the labels. We can directly see here the main advantage of clustering in deep learning. Creating the labels can represent a quite burdensome task. Moreover, the labeling process has the risk of introducing biases in the data. Especially in the case where the labeling process is accomplished by the humans, there is a serious risk of introducing biases related to human behaviors in the data.

Clustering is broadly divided into two categories of algorithms: partitional clustering and hierarchical clustering. Partitional clustering consists of partitioning (separating) the data into different non-identical groups. One of the most popular algorithm in this category is the famous k-means clustering [4]. Different groups of data are constructed using some similarities or even dissimilarities measures.

Hierarchical clustering [5] consists of constructing the clusters in a recursive fashion, using the previously established clusters. This recursive technique can be accomplished in two different ways. The first one is the bottom-up manner, where every single data is first considered as a cluster itself before a recursive merging process is conducted in order to merge clusters that are identical. The second way is the top-down manner, where the totality of the data is first considered as one single cluster before a recursive splitting process is conducted in order to divide the first big cluster into smaller clusters.

Two big drawbacks can be noted for these conventional clustering methods. The first limitation is that, as for their counter-parts in the supervised learning paradigm, they all rely on the discriminability of the features. As a normal pattern recognition technique, discrimination in the conventional machine learning models, whether it is classification or clustering, comes after the most important part of the process: the feature extraction. Fig. 2 shows an illustration of the pattern recognition framework. As said before, the first limitation of the afore-mentioned clustering methods is that their performance depends exclusively on the extracted features. If the features are easily discriminable, we can expect high performance but, otherwise, the clustering performance will be highly limited.

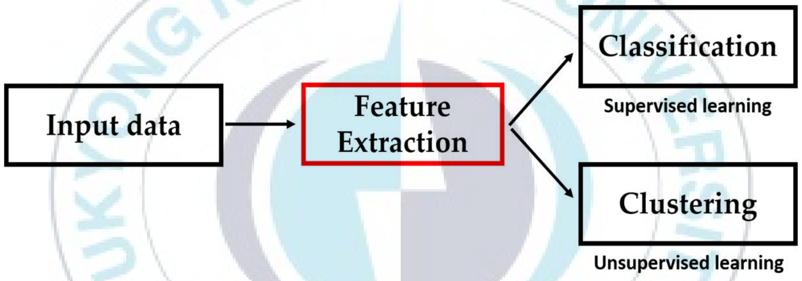


Fig. 2. Pattern recognition framework.

This situation can be explained by the second drawback of these conventional clustering methods. Their learning is static. Which means that the feature space does not change during the learning process, only the clusters change. The features are fixed, precisely because they were learned before the clustering process.

Deep learning offers the possibility of automatic feature learning process. In general, deep learning structures are mostly well suited for classification (i.e. supervised learning) tasks. Clustering with deep learning mostly invokes what we call dimensionality reduction. Dimensionality reduction consists of reducing the dimensions of the original data in a shorter, more precise and less redundant form. Besides deep learning, popular dimensionality reduction methods include principal component analysis (PCA) [6] or t-distributed stochastic neighbor embedding (t-SNE) [7]. With deep learning, reducing the dimensions of the data can be accomplished by some structures that map the original data into some spaces, usually denoted as latent spaces, before mapping them back to their original input space.

Structures that perform this type of operation are called autoencoders (AEs) [8-10]. They comprise two parts: the encoder that compresses the dimensions into a smaller one, and the decoder that uses the compressed version of the data and transforms it back to the original form. We can say that the encoder maps the data from their original input space to the latent space while the decoder maps the features from the latent space back to the input space. Stacked autoencoders (SAEs) [8,11] denote the structures where many autoencoders are "stacked" on top of each other in order to form a sufficiently deep architecture.

Another variant of autoencoders is the denoising autoencoder (DAE) [12]. Unlike with normal AEs, where an identity learning is performed, the DAE takes a noisy version of the input and learns how to reconstruct the original clean data. AEs and DAEs perform learning using the same loss function, which goal is to minimize the differences between the original data and the reconstruction. The only difference is that AEs take as input the original data while DAEs take the corrupted version of the input. An illustration of the DAEs is depicted in Fig. 3. In Fig. 3, $\boldsymbol{X}$, $\widetilde{\boldsymbol{X}}$ and $\boldsymbol{Y}$ denote, respectively, the input data, the corrupted version of the input and the latent features.

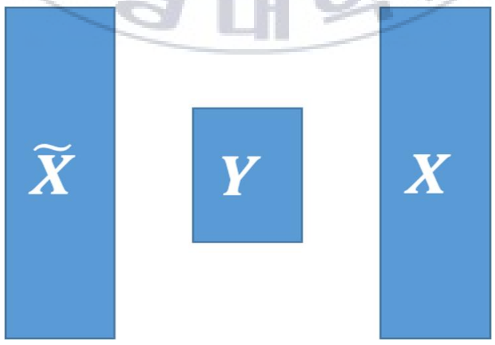


Fig. 3. Illustration of the denoising autoencoder.

Another variant of AEs is the sparse autoencoder [13]. It has the same structure with AEs with the only difference that the features that it should learn must have the property of sparsity. Sparsity denotes the constraint that most of the units in the hidden layer (the latent features) should be inactive, i.e., they should be zero.

Another popular structure of autoencoders is the deep convolutional autoencoder (DCAE) [14]. As suggested by Guo et al [15], DCAE is well suited for images because it has the capability of avoiding the distortion of the local structure of the original images, unlike AEs and its variants. DCAE also uses the encoding-decoding process with the advantage of utilizing the two-dimensional structure of the convolutional neural network (CNN). An illustration of the DCAE is depicted in Fig. 4, where we can clearly distinguish the encoder and the decoder.

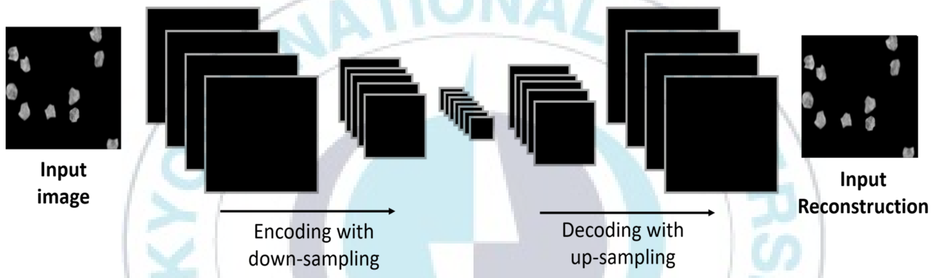


Fig. 4. Illustration of the deep convolutional autoencoder (DACE).

Deep learning-based clustering mostly consists of two steps. In the first step, as part of the feature extraction process, the input data is reduced in the lower dimensions by using one of the structures described above. In the second step, as part of the discrimination, one of the clustering methods previously described is utilized in order to construct the clusters. Even though this approach utilizes efficiently the automatic feature learning mechanism afforded by deep learning, the principal drawback is the same with the conventional clustering methods: the learning still remains static. As we can clearly understand, the feature extraction is performed separately with the clustering. Consequently, there is no direct link between the features that are learned and the clustering that is performed later. The process denoted by deep clustering tries specifically to solve this problem of disparity between the feature learning and clustering processes. The next section introduces the basic concepts of deep clustering.

2.2 Deep clustering

Deep clustering regroups the methods that try to perform dimensionality reduction and clustering at the same time. This direct connection between the two processes allows the adopted deep learning structure to explicitly learn features that can be easily clustered. Furtherly, as opposed to the static learning mentioned in the previous section, this connection between the two processes triggers a learning that can be qualified as dynamic. Dynamic in the sense that the features used for clustering change every time according to the feedbacks of the previous iterations. We can separate these kinds of approaches into two distinct groups: pseudo-classification methods and deep embedded clustering methods.

The particularity of the pseudo-classification methods is that they utilize CNN in the same way as used by the supervised learning methods, with the big difference that the labels used for updating the CNN's parameters come from the clustering process. Different works that adopt this approach can be found in [15-19]. We named these kinds of methods as pseudo-classification because they mimic the concept of classification by minimizing the differences between the network's outputs and the "pseudo-labels" generated by the clustering algorithm, similarly as done in the classification works.

One of the state-of-the-art works that uses this approach has been proposed by Caron et al. [20] where the authors have connected a clustering module in top of the output layer of a CNN. Normal cross-entropy loss is used in order to minimize the differences between the CNN's outputs and what they have called "pseudo-labels". These pseudo-labels are precisely the cluster's assignments computed using the k-means loss function. In other words, their method alternately computes the cluster assignments with k-means by using the output values of the CNN and subsequently updates the CNN's parameters by using the previously computed cluster assignments as the labels. We have used this method in our comparative study as part of the pseudo-classification approach.

The second group of works utilize the different types of autoencoders described in the previous section. These methods are called deep embedded clustering because they embed a clustering layer in the middle of the autoencoders whose purpose is precisely to cluster the latent features learned during the training. At every iteration, the clustering layer performs clustering on the features produced by the autoencoders. This type of approach is used in different manners by Yang et al. [17] and Guo et al. [15], the first for the clustering of randomly generated data and the latter for the discrimination of the images representing the handwritten digits. The authors in [17] have embedded a clustering layer in the SAEs, while the authors in [15] have utilized the DCAE. Other works using similar idea can be found in the interesting review made by Min et al. [19].

Of great interest, Dizaji et al. [21] have utilized a dual DCAE. Two different encoders are used in their work: the first, called "clean encoder", takes the original images as inputs and the second, denoted as "noisy encoder", takes the noisy version of the inputs. Both the clean and noisy encoders are connected to a single decoder whose purpose is to learn how to reproduce the original inputs. This work represents one of the state-of-the-art of the second group of deep clustering methods. We have used this work for the comparative study as part of the deep embedded clustering methods.

Our proposed deep clustering method belongs to this second group, because we have also adopted the DCAE as our deep learning model. One interesting point about these afore-mentioned embedded clustering methods is that the reconstruction process is not really taken into account. Even though interesting discussions are provided in [15] about how to minimize the distortion of the local structures of the original images, the reconstruction task is performed only by the DCAE's loss. The main contribution of the present work is to propose different architectures designed specifically in order to tackle the problem of the reconstruction performance.

As mentioned in the introduction, the second method of this work utilizes some parts of the field of explainable artificial intelligence. The next section presents the concept behind explainable AI.

2.3 Explainable artificial intelligence

2.3.1 Overview

Despite the fact that the deep learning-based models have achieved tremendous performances in various areas, it has been reported that they have not been significantly deployed in the clinical area. The reason beyond that is the “black-box” nature of those models [22]. The meaning behind of the term “black-box” here is the fact that it is generally difficult to clearly explain the decisions made by the deep learning models. By comparison, machine learning models like decision trees are easily explainable since we can visualize progressively the decisions made by the model by exploring the graph representing the tree.

In medical areas, explainability is crucially important because a medical diagnosis needs to be transparent and understandable in order to gain the trust of the physicians as well as the patients. Explainability here refers to the fact that the users, in this case the physicians and the patients, should be able to fully understand the underlying mechanisms that make a given artificial intelligence (AI) system to produce a given decision (in this case, a medical diagnosis). The black-box nature of deep learning models does not permit such explainability. That is the reason why many researchers in the past decade have been working on many different ways of making deep learning models, and AI systems in general, to be explainable. All of these works are regrouped in the area called explainable AI (XAI). A detailed analysis of the terminologies, concepts and usage cases of XAI is provided in the work by Arrieta et al. [23].

As schematically illustrated in Fig. 5, the question to answer by a given XAI method is the reason why the deep learning model gives any specific output. In the example shown in Fig. 5, where the input image represents a cat, the deep learning model correctly outputs the class “cat”. For instance, the question that will try to answer an XAI algorithm is the reasons that motivated the model to produce this particular output.

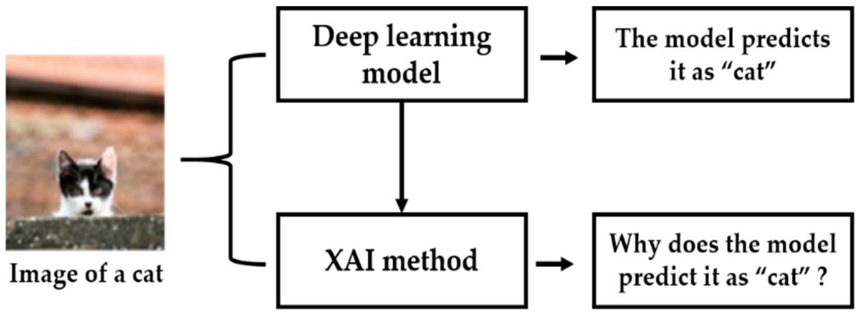


Image of a cat
Deep learning model
The model predicts it as "cat"
XAI method
Why does the model predict it as "cat" ?

Fig. 5. Schematic representation of XAI methods.

Various different methods can be used in order to explain the decisions made by the models. Some of these methods are based on expert knowledge [24], interactions with the user using textual justifications [25], or "similar images"-based explanations [26], where similarly labeled images are provided to the user as a reason for making a particular prediction for a given image.

The group of the methods that are of a great interest for us in this study is the group called "attribution-based method". This ensemble of methods tries to produce a map, called "attribution or saliency or relevance" map (in this study, the term relevance map is preferably used), that shows the most important features that were used by the model in order to predict a given class. In other words, relevance maps designate the maps that show, in case of images, the pixels that contributed the most to the output of the model. In case of the example shown in Fig. 5, the XAI algorithm must produce a map of the pixels that contributed the most to the output "cat". To the question "why does the model predict it as "cat"? The relevance map gives the following response: "because of these particular pixels that are located at these particular positions".

As reported by Singh et al. [22], a majority of the works in the medical imaging literature that are focused on explainability opts for this kind of explanation, i.e., they utilize the relevance maps as the principal explanation of the decisions made by the deep learning models. In the next sub-section, we discuss about some of the methods that can help to produce the relevance maps.

2.3.2 Relevance maps

The goal of a relevance map is to assign a contribution (relevance) value to all of the features that led the network to produce a given output. One way of producing these maps is the fact of performing perturbation on the input image. This idea comes from the hypothesis that every single input feature singularly contributes to the final output of the model. Thus, removing a feature in the input should consequently affects the output of the model. In this scenario, evaluating the effect of a given feature on the output of the model can just consist of removing the feature and analyzing how this removal affects or changes the final output of the model. Contributions are assigned according to the amount of changes that occurred after the removal of the feature. In other words, features are said to have a big contribution if their removal from the input vector causes a big change in the output of the model. On the contrary, features are said to be "irrelevant" if their removal does not cause much of change in the output. In practice, the features will be ranked according to the amount of changes caused by their removal.

This kind of method can be implemented by removing, masking or modifying certain input features, then by running the forward pass, which consists of giving the modified input to the model, and finally measuring the changes that occurred to the output. This technique was primarily applied by Zeiler and Fergus [27] to the CNN. We can easily understand that this method is computationally expensive. As reported in [22], occluding all the input features one-by-one and running the forward pass every time after a particular occlusion takes a huge amount of time.

Another way of producing relevance map consists of back-propagating the relevance from the output layer to the first layer, which contains the input vector. This idea is the most used in the relevance map literature. The works utilizing this idea are called "backpropagation-based methods" and many of them can be found in [28-33].

Two backpropagation-based methods have our particular interest. The first one is the layer-wise relevance propagation (LRP) proposed by Bach et al. [34]. Using the idea explained above (backpropagation), the LRP algorithm first runs a forward pass using a given input in order to produce the output. In the second step, this output is used as the starting point of a propagation that goes layer after layer until we reach the input layer. At the input layer, we can have the contribution values of every single feature. Readers are encouraged to refer to [34] for further detailed explanations of the LRP algorithm.

Examples of the relevance maps produced by this method are shown in Fig. 6. Fig. 6 (a) shows an example image of a nucleolar with a very low contrast (HEp-2 images with this kind of contrast are called “negative intensity” image. Details about this situation are developed in section 3.2.2) and Fig. 6 (b) shows the relevance map produced by the LRP technique for this particular image. We can remark how the most important characteristics of this cellular type are clearly exposed, even though they are not very clear in the original image because of the low contrast. In terms of relevance, pixels that have strong contributions are shown in red, with the strongest ones being in yellow, while the “irrelevant” pixels are shown in black.

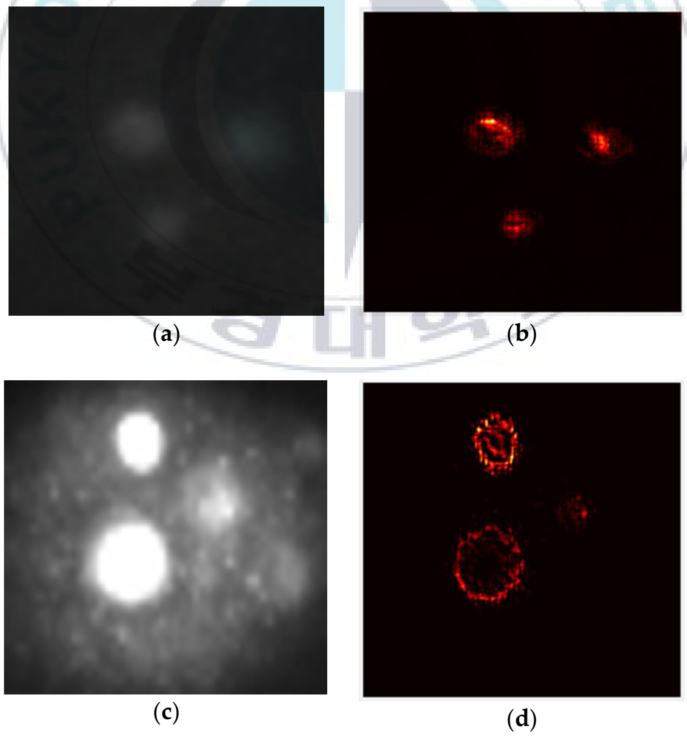


Fig. 6. Examples of relevance maps produced by LRP: (a) a nucleolar image with poor contrast, (b) its relevance map, (c) a nucleolar image with strong contrast, and (d) its relevance.

Fig. 6 (c) shows a nucleolar with a strong contrast (positive intensity HEp-2 image). In Fig. 6 (d), we see the relevance map produced by the LRP for this image. Again here, we can notice that most of the important features of this HEp-2 cell type are highlighted by the relevance map. Note that all of these maps were produced using our deep parallel residual networks that will be presented in details in section 3.2.2.

Another backpropagation-based method that has our interest is the DeepTaylor algorithm proposed by Montavon et al. [35]. As for the first method described above, this technique also utilizes the forward pass in order to produce the network's output and then runs a backward pass from the output layer to the input layer in order to produce the relevance scores for every single feature. This method has the particularity of using the Taylor decomposition (that is the reason why the authors have named their method as DeepTaylor) in order to compute the relevance scores. Readers are also encouraged to refer to [35] for further explanations of this method.

Examples of this relevance map method are shown in Fig. 7. Fig 7 (a) and Fig. 7 (c) both describe the same images shown in Fig. 6 (a) and Fig 6 (c). Fig. 7 (b) shows the relevance map produced by the DeepTaylor method for the negative intensity nucleolar image and Fig. 7 (d) shows the relevance map produced by the same method for the positive intensity nucleolar. We can see that, compared to the relevance maps produced by the LRP (Fig. 6), the DeepTaylor maps in Fig. 7 display much more important features. Beside these differences, we can notice that the most relevant features (pixels that tend to be yellow) are mostly located at the same positions for the two methods.

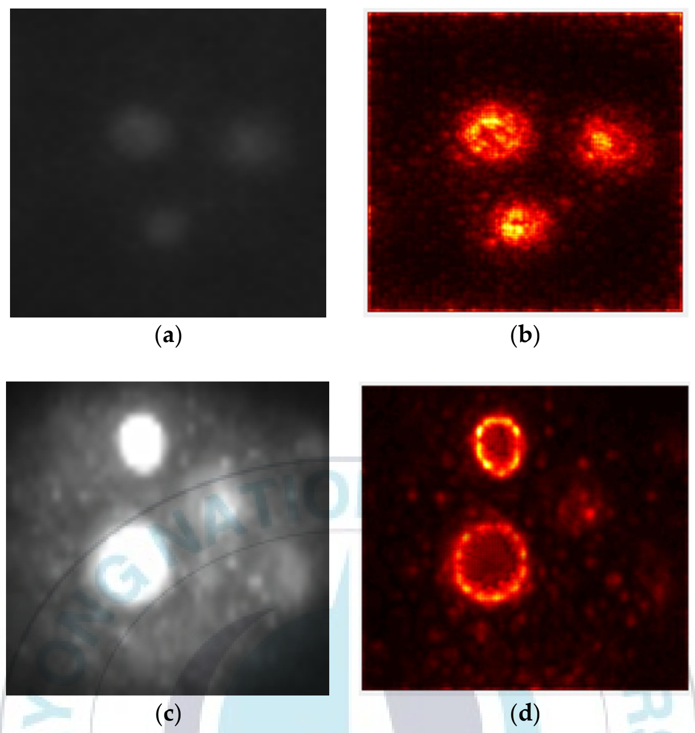


Fig. 7. Examples of relevance maps produced by DeepTaylor: (a) a nucleolar image with poor contrast, (b) its relevance map, (c) a nucleolar image with strong contrast, and (d) its relevance.

As previously said, we will use these relevance maps in the formulation of our active learning method. But, before going any further, the question that comes in mind about these maps is: how can we evaluate them? How can we know for sure that the features pointed by the maps are really relevant? The next section will briefly discuss about the evaluation of the relevance maps.

2.3.3 Evaluation of relevance maps

The question of the evaluation of the relevance maps is one of the important topics discussed in the XAI literature nowadays. Even though this question remains challenging today, a trivial solution can easily come in mind from the definition of relevance map given in the previous section. In fact, researchers have defined a relevance map as the map that shows the features that contribute the most to the model's output. A trivial solution for evaluating the relevance maps can just be to remove those features that are said to be relevant and then see how this removal changes the behavior of the model. Intuitively, this solution postulates that if a feature is important, its removal should sufficiently impact the output.

Samek et al. [36] were the first to propose this idea in order to evaluate the relevance maps. Their idea stipulates that a relevance map can be seen as an ordered sequence. In fact, as explained before, methods like LRP and DeepTaylor assign a relevance score to every single feature. Those scores can be ordered from the most relevant to the least relevant in order to produce an ordered sequence.

Their idea, called "most relevant first", consists of masking (by applying some perturbations) the features by following the order of their relevance, so that, the most relevant features are masked (or removed) in priority. And then, the next step is to give the modified data to the network and see how these perturbations impact the output of the model. If the impact is significant, we can conclude that the relevance map really showed the most important features of the data. If the impact on the output is insignificant, we can conclude the contrary. The expected impact here is the degradation of the model's performance. The authors in [36] postulate that if we remove the most relevant features of the data and give those modified data to the model, we should expect a serious degradation of the accuracy.

As pointed by Hooker et al. [37], this perturbation-based evaluation scheme poses certain problems. The most important problem is that machine learning frameworks in general are based on the hypothesis that the training data (seen during the fabrication of the model) and the testing data do share some similar distribution. When we remove some parts of the original data, their distribution necessarily changes, an effect denoted as "shift in distribution" in [37]. The question is then: how can we be sure that the performance degradation comes from the information removal (the fact of removing the most relevant features) and not from the shift in distribution effect?

This question is crucial since the evaluation of the relevance map is based on the fact that the model's performance should decrease after we remove the features that were designated as relevant. In order to solve this problem, Hooker et al. [37] proposed what they have called "ROAR", for remove and retrain. The idea is simple: after removing the most relevant features, we should retrain and re-test the model by using the perturbed dataset. In this case, and only in this case, if the performance decreases, it will not be because of the shift in distribution but because of the information removal.

Still in [37], another experiment is called "keep and retrain" (KAR), where we do the opposite: we keep the features that were pointed as relevant, and remove the irrelevant ones and then retrain and re-test the model. In this case, the model's performance is expected to not decrease because, precisely, only the most relevant features were kept in the data and the irrelevant ones were removed.

As said before, the formulation of our active learning method includes the use of the relevance maps. We have used the ROAR and KAR setups in order to select the best relevance map methods for our networks. Results of these experiments are presented in details in section 3.2.4. Readers that are really interested in this topic can find another evaluation technique, called "sanity checks", proposed by Adebayo et al. [38]. In the next section, we present succinctly the basic ideas behind active learning.

2.4 Active learning

Active learning aims to alleviate the labeling process by allowing to select only the data that should be given to the learning model. The goal is to find which are the data that carry the most informative details that can help us to build the classifier. This selection can be done by finding the data for which the model is the most uncertain. This is referred as uncertainty sampling [39]. The idea is to find an uncertainty measure that can help to evaluate the confidence of the model and then use that measure in order to select only the data for which the model is the most uncertain. Select in order to annotate them. Instead of annotating (labeling by hands) all the available data, we can just annotate the ones for which our model is the most uncertain about.

Different uncertainty measures have been utilized in the literature. Among the most used, we have the entropy [40], which can measure the certainty level of a classifier by using the probabilities (scores) that are attributed to each class by the classifier. For a given instance x, the entropy can be evaluated by the following equation:

$$entropy(x) = \sum_{j=1}^{N} p(y_j|x) \log p(y_j|x), \tag{1}$$

where the values $p(y_j|x)$ represent the classes' probabilities (or classes' scores) as outputted by the classifier for the data x and N is the number of classes. When the entropy is very high for a given data, it means that the classifier evaluates equally the different classes for that data. In other words, the classifier is uncertain about which class to assign to the input data x.

Another way to measure the uncertainty of a model is to compute its amount of confidence for a given instance. This method is referred as least confidence-based sampling [39,41]. For a given data x, the confidence C is given by

$$C = \underset{j}{\operatorname{argmax}}\, p(y_j|x), \tag{2}$$

where the values $p(y_j|x)$ represent also the classes' probabilities. We know that for every single instance, the classifier outputs a vector containing the probabilities associated to each class, represented here by $p(y_j|x)$. The confidence denoted in Equation (2) finds the maximum score. Which means that it finds the class for which the model assigns the maximum probability value. Which gives us the amount of confidence of the model for every single data. The idea is to select the data for which the model is the least confident, which means the data for which the values C are the lowest. In practice, we can sort the data according to their confidence C from the smaller to the larger, and then prioritize the annotation in that manner. Note that entropy also can be thought of as the amount of confidence. When the entropy is high, it means that the confidence is low and vice versa.

Another method, called the margin sampling [42], consists of computing the difference between the highest score and the second highest score in the vector of probabilities. In this case, the confidence C is given by

$$C = p_1(y_j|x) - p_2(y_j|x), \tag{3}$$

where $p_1(y_j|x)$, and $p_2(y_j|x)$ are the highest and the second highest probabilities, respectively. Similar with the previous case, the data with the lower C values are prioritized for the labeling. The difference computed in Equation (3) enquires us about the confidence of the model. If its value is high, it means that one of the class has a much bigger probability compared to the other and, on the contrary, when its value is small, it means that the model evaluates equally the two classes, which means that the model is uncertain.

The least confidence and margin sampling methods work similarly and, unlike the entropy, are both less suitable for the multi-class classification. In fact, in Equation (3), only two classes are taken into account, while the entropy, depicted in Equation (1), utilizes all the classes' probabilities. Other uncertainty sampling methods can be found in [43-45].

The selection process in conventional active learning is performed through an iterative process depicted in Fig. 8. The different steps listed in Fig. 8 are explained in the following points:

(1) The first step consists of having an initial set of labeled data. These data are usually selected randomly among the all of the unlabeled data.
(2) Use the initial labeled data in order to train our model.
(3) Use the trained model on the unlabeled set in order to generate the probabilities for every single data.
(4) Compute the confidence with the probabilities by using one of the methods described above.
(5) Select the data with the lowest confidence and annotate them.
(6) Mix the newly annotated data with the initial labeled set. Get back to step (1).

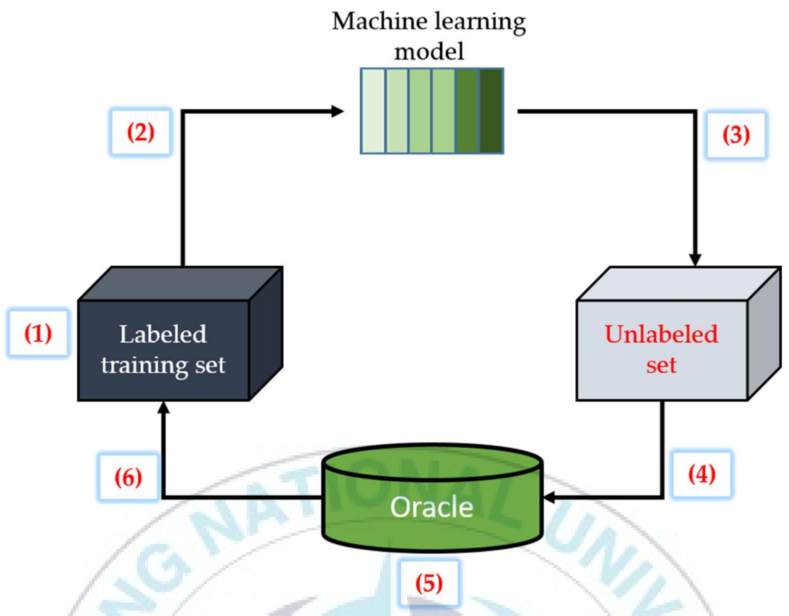


Fig. 8. Schematic representation of the conventional active learning method.

This iterative process can continue until we annotate the totality of the data. If the goal is to alleviate the labeling process, the iterative labeling can stop when the labeling limitation, called “sampling budget”, is reached.

As we can clearly see in the different active learning steps described above, the selection process is based on the model’s output. In fact, in step (4), the confidence is measured by using the probabilities outputted by the machine learning model. In this work, we propose a new selection method for active learning. Our selection will utilize the relevance maps.

The next section presents briefly some of the works that utilize unsupervised learning methods for the medical image analysis.

2.5 Unsupervised learning in medical images analysis

As discussed previously, the majority of the methods in the literature prefer to adopt the supervised learning paradigm for the applications using deep learning in medical images analysis. However, since the past two decades, several works tried to propose unsupervised learning approaches for the medical images. Most of these works utilize the SAEs and some of them prefer to adopt the CAEs, as they are more suited for images. It should be noted that, all of these methods propose a feature extraction process that is strictly unsupervised. SAEs and CAEs provide end-to-end learning for building features in an unsupervised learning way.

Nonetheless, for the majority of them, the discrimination part is conducted in a supervised learning way. Most of the methods propose to use the features learned by the SEAs or the CAEs as the inputs of a nonlinear classifier in a second step. Some works, for the discrimination part, prefer to adopt one of the clustering methods described in section 2.1.

Different methods that utilize the SAEs for the classification of the Alzheimer's disease using the images produced by the magnetic resonance imaging (MRI) system can be found in [46-48]. Different others methods for classification or segmentation of medical images utilizing the SAEs can be found in [49-52].

As described in section 2.1, another popular version of the SAEs is the sparse autoencoder. Stacked sparse autoencoders (SSAEs) have also been utilized in the area of medical images analysis. As an example, Xu et al. [53] have used the SSAEs for the detection of nuclei on breast cancer images. Different other methods that use the SAEs for different types of classification tasks can be found in [54-56] and for segmentation, in [57,58]. For the different other methods, users are encouraged to refer to [59-66].

We mentioned in the introduction that our target data are the HEp-2 cell images. In the next section, we present a quite detailed review of the different methods, from the handcrafted features to the deep learning-based ones, that are used in the HEp-2 cell classification literature.

2.6 HEp-2 cell classification in literature

As mentioned before, the classification of the different types of the HEp-2 cells is one of the most important steps in the diagnosis procedure of autoimmune disease. Performing this classification manually represents an arduous task and can cost a lot of time during the diagnosis process. Moreover, the manual analysis of the HEp-2 cell patterns poses a certain problem in terms of consistency of the diagnosis results, since the complexity of the images complicates the task for the pathologists. This is the reason why the automatic discrimination of the different types of the HEp-2 cell images is more than necessary in order to help pathologists during the diagnosis procedure. Which makes the classification of these cells to be one of the important parts of the computer-aided diagnosis systems.

Different methods have been presented for this task in the literature. As a pattern recognition problem, the classification of the HEp-2 cells is usually tackled with a feature extraction part followed by a discrimination process. Feature extraction consists of extracting (or selecting) the information that is supposed to help differentiating the different cellular types. The second part of the process consists of utilizing the obtained features as the inputs of a discriminator (a classifier). Different hand-crafted features have been proposed for this purpose and many of them can be seen in the review by Foggia et al. [67]. Various descriptors like the discrete cosine transform [68,69], the scale-invariant feature transform [69,70], the local binary patterns [71-73] or many other different statistical features have been highlighted in the literature [74-79]. A multi-class support vector machine (SVM) is mostly used as the discriminator for these methods.

The automatic feature learning process afforded by deep learning has largely supplanted the use of these hand-crafted features. In addition to the fact that the subjective choice of the features was a disadvantage for these methods in terms of consistency, their limitations in terms of the discrimination results explain why they have fallen out of use and been supplanted by the deep learning-based methods. In fact, nowadays, the quasi majority of the works in the literature utilize these methods in order to demonstrate their superiority over the conventional handcrafted features.

One of the pioneers works to adopt the CNN for the HEp-2 cell classification was the method proposed by Foggia et al. [2] at the International Conference on Pattern Recognition (ICPR) HEp-2 cells classification contest in 2012. Since then, multiple works have proposed the use of CNN models in many different ways [80-85]. Among the most noticeable, Li and Shen [86] have presented a customized CNN model, called the deep residual inception network (DRI-Net), which associates the residual connection from the ResNet [87] and the “Inception modules” utilized in the GoogleNet [88]. Also among the most noticeable, Shen et al. [89] have used the ResNet approach but with a much deeper residual module with several cross connections between the layers. Their model was named the deep-cross residual network (DCR-Net) and they have tested a huge data augmentation process in order to boost the classification accuracy.

Interestingly, Majtner et al. [90] have proposed the use of generative adversarial networks [91], the deep convolutional generative adversarial networks specifically (DCGAN) [92], in order to generate realistic artificial HEp-2 images. The goal was to augment the existing datasets with the artificial images generated by the DCGAN. Li and Shen [93] have extended the idea presented in [86] by enlarging the convolutional kernels’ size and adding more convolution operations with different dilations within the DRI modules. The short-cut connection is made outside the DRI module, which means that the residual connection is made between the input and the final output of the module, while in the previous version [86], different residual connections were made between the layers inside the DRI module.

Transfer learning consists of using an already trained network for a new task. Fine-tuning the trained network consists of updating its parameters using the new dataset. This technique was used for the HEp-2 cells classification by Phan et al. [94], who utilized a model, the VGG-16 network [95], that was previously trained on the ImageNet dataset. Some others methods utilize a pre-trained network only as a feature extractor. The high-level features extracted from the pre-trained CNN model are then used in order to train a multi-class SVM. In the HEp-2 cell classification literature, this technique (transfer learning without fine-tuning) has been adopted in different ways by the works discussed next. Lu et al. [96] have used a pre-trained VGG-16 network as a feature extractor while Nguyen et al. [97] proposed the use of an ensemble of networks. Another work using this idea was presented by Cascio et al. [98] with the use of the AlexNet [99] as the feature extractor. For all of them, the discrimination was performed by an SVM, or by both an SVM and a k-nearest neighbor classifier, as done in [98].

An interesting transfer learning approach, named cross-modal transfer learning, was proposed by Lei et al. [100]. Cross-modal transfer learning consists of updating the parameters of the pre-trained network (fine-tuning) in two steps: first, by using a quite small dataset, then, second, by performing the update on the targeted dataset, which is supposed to be much larger and more complex than the first one. With the particularity that the two datasets have to be similar, which means that they have to share the same feature domain. In fact, most of the pre-trained CNN models were trained on the ImageNet dataset, which contains images that are far different with the HEp-2 cell images. The idea of cross-modal transfer learning is like performing a pre-fine-tuning (on the small dataset) before a final fine-tuning (on the targeted dataset) in order to smooth the parameters' updating process during the training. The authors in [100] have used ResNet-50 as the network to be fine-tuned. The small dataset utilized was the ICPR2012 dataset [2], and the targeted dataset was the ICPR2016 dataset [101], also known as the 13A dataset in the literature (in the remaining part of this work, the denomination 13A dataset is preferably used).

The most important point to understand about this review is that the researchers in this field, both for the handcrafted features-based and deep learning-based methods, as described above, prefer to adopt, quasi unanimously, the supervised learning paradigm in order to tackle the HEp-2 cell images classification task. As mentioned before, the supervised learning-based methodology necessitates the presence of labeled images. And, even though the discrimination performance that we can obtain by using this methodology remains remarkable, the necessity of constructing labeled datasets that contain a considerable number of images represents a serious concern. In fact, deep learning-based methods require the presence of thousands of images. And, the process of labeling by hands these images can eventually represent a quite onerous task in the future, when we will have to create more expanded datasets, which can be a drawback for this methodology.

On the contrary, our work aims to propose both an unsupervised learning and semi-supervised learning approaches for the HEp-2 cell classification task. The next chapter presents in details our proposed methods. The first section (section 3.1) presents the unsupervised learning method that is based on the deep clustering concept. The second section (section 3.2) will present the semi-supervised learning approach that utilizes the concept of active learning.

III. PROPOSED METHODS

3.1 Proposed deep clustering method

3.1.1 Method overview

As said before, in our previous work [3], we have investigated the unsupervised learning scheme for the HEp-2 cell classification. But, only the feature extraction part was completely done in an unsupervised learning way. The created features were furtherly given to a nonlinear classifier that learns with the use of the manually labeled data. Which means that, while the feature extraction was done in an unsupervised way, the discrimination part had still remained supervised. The first contribution of this present work is to propose a scheme where the feature extraction and the discrimination parts are all performed in a strictly unsupervised paradigm.

We adopt the deep convolutional autoencoder (DCAE) as the principal feature extractor. The DCAE takes the original images as inputs and learns how to reproduce them via an encoding-decoding structure. Unlike in our previous work, where the features learned by the DCAE were extracted and then utilized as the inputs of a nonlinear classifier, we propose to embed a clustering layer in the DCAE. The clustering layer will learn, in every single step of the training process, how to automatically discriminate the latent representations produced by the DCAE. As discussed in section 2.2, this technique is called deep clustering and we have presented different works that utilize it.

Unlike in [15], where the efficiency of the reconstruction process performed in the DCAE's decoder is assured only by the model's loss function, the principal contribution of the present work is to utilize some techniques in order to minimize the loss of the spatial information of the original input images and, thus, to ensure a certain preservation of the local structure of the original pixels, which will probably ensure a better reconstruction. In fact, the down-sampling process incorporated in the DCAE and performed by its encoder causes the loss of the spatial details inside the network. As a contribution, we investigate in this work how the quality of the reconstructed images can affect the quality, thus, the discrimination potentiality, of the latent representations learned by the DCAE.

We have investigated the effectiveness of the proposed deep clustering method on two benchmark datasets of the HEp-2 cell classification, the 13A dataset and the SNPHEp-2 Cell dataset [69]. The obtained results demonstrate that the proposed strictly unsupervised learning method outperforms by far the handcrafted features and performs at least at the same level with the state-of-the-art supervised deep learning methods. The schematic illustration of the proposed method is shown in Fig. 9.

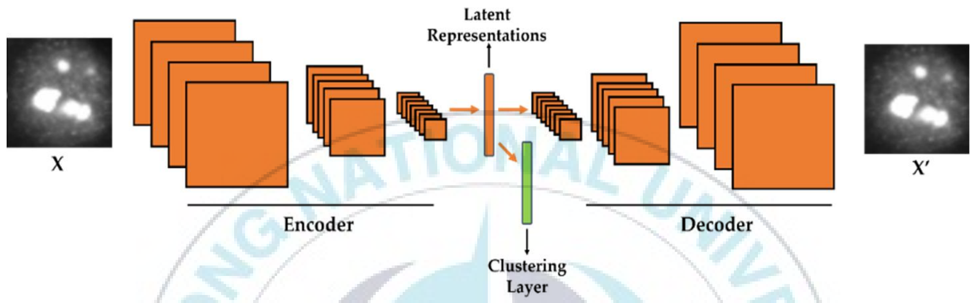


Fig. 9. Illustration of the proposed method. A clustering layer is embedded in the convolutional autoencoder in order to learn how to cluster the latent representations. X is the original cellular image and X' is the reconstruction.

3.1.2 Deep embedded clustering network

Autoencoders refer to unsupervised learning-based structures that are mainly used for the purpose of dimensionality reduction. Dimensionality reduction, which can also be utilized as a feature extraction, consists of finding a better representation of the data in lower dimensions. Deep neural network (DNN)-based autoencoders consist of two principal structures: the encoder and the decoder. Given an input signal **x**, with $\mathbf{x} \in \mathcal{R}^D$, the encoder takes it and transforms it into a contracted representation **y**, with $\mathbf{y} \in \mathcal{R}^d$, with $d < D$, by utilizing the transformation function g in such a way that

$$\mathbf{y} = \mathrm{g}(\boldsymbol{\theta}\mathbf{x}), \tag{4}$$

where $\boldsymbol{\theta}$ englobes all the different parameters of the encoder, which can be a set of weights and biases. After the encoder transforms the input signal **x** into **y** using Equation (4), the decoder takes the contracted representation **y** as input and uses the same transformation function g but, in this time, for the purpose of reconstructing the original signal $\mathbf{x}$. Here, let $\mathbf{z}$ be the output of the decoder. Then, we have

$$\mathbf{z} = \mathrm{g}(\boldsymbol{\theta}'\mathbf{y}), \tag{5}$$

where $\boldsymbol{\theta}'$ englobes all the different parameters of the decoder, which can also be a set of weights and biases. Finally, the network, composed of both the encoder and the decoder, should learn the parameters $\boldsymbol{\theta}$ (encoder) and $\boldsymbol{\theta}'$ (decoder) in such a way that the reconstructed signal $\mathbf{z}$ equals to the input vector signal $\mathbf{x}$. Which means that the network should learn the parameters that help minimizing the most the existing differences between the input $\mathbf{x}$ and the final network's output $\mathbf{z}$.

In case of images or signals that are represented in a two-dimensional (2-D) fashion, this described encoding-decoding process can be realized with the use of another DNN-based structure, the so-called deep convolutional autoencoder (DCAE). Because it takes 2-D signals, the DCAE is likely to be more efficient than the SAEs as far as images are concerned. In the DCAE, the encoder performs the down-sampling process, while the decoder will perform the opposite operation, the up-sampling (See Fig. 9 for the illustration of the encoding-decoding process). The down-sampling can be realized with the use of the convolutional and/or pooling layers. On the other hand, the up-sampling is done by the backwards convolution (transposed convolution), often called "deconvolution", and/or by the backwards pooling, often denoted as "unpooling" operations. The final solution of the DCAE is given by

$$(\boldsymbol{\theta}, \boldsymbol{\theta}') = \operatorname*{argmin}_{\boldsymbol{\theta}, \boldsymbol{\theta}'} L(\mathbf{xz}), \tag{6}$$

where $\mathbf{z}$ denotes the reconstruction (decoder's output), $\mathbf{x}$ represents the original image (encoder's input) and the function $L(\cdot)$ represents the cost function that measures the differences between $\mathbf{x}$ and $\mathbf{z}$. In this work, the adopted cost function is the squared Euclidean distance described as

$$L(\mathbf{xz}) = \sum_{i=1}^{N} \|\mathbf{x}_i - \mathbf{z}_i\|_2^2 \tag{7}$$

where the value N represents the total number of data. The network will learn the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$ by minimizing the cost function represented in Equation (6).

Some of the early works that aimed to incorporate the clustering process in the DNN can be found in the methods previously cited in section 2.2. For example, the authors in [16] and [17] have proposed the idea of connecting a clustering module to the output layer of a DNN and to learn the DNN's parameters and the clusters' centers at the same time. In their works, only the clustering loss is used during the parameters' updating process, as the reconstruction is not a concern. And the fact that they have adopted the plain network, the SAEs, which takes one-dimensional data as inputs, poses the problem of the preservation of the local structure of the pixels of the original images.

Guo et al. [15] have proposed to use the DCAE, instead of the SAEs, with the clustering layer incorporated in the middle of the network, instead of connecting it to the output layer, as done in [16,17]. But, in their work, the reconstruction problem is solved only by adding a reconstruction loss to the clustering loss. In our work, as we will explain later, in addition of using a reconstruction loss, we also apply some techniques that can assure a better reconstruction and investigate how it can affect the quality of the latent representations.

Yang et al. [102] have utilized the SAEs and embedded a clustering module (or layer) in the middle. They have used a global loss function that incorporates the reconstruction and the clustering losses. In their work, the k-means clustering cost is used as the clustering loss. In general, the global loss function L for the DCAE can be expressed as

$$L = L_r + \gamma L_c, \tag{8}$$

where L_r is the reconstruction loss and L_c denotes the clustering loss. The parameter γ is used to balance the importance of the clustering loss in the global loss. If γ is greater than one, the clustering loss will affect more the global loss, otherwise, the reconstruction loss will have more incidence during the training. The reconstruction loss L_r is defined to minimize the differences between the output of the decoder, which is the final output of the network, and the original input image. This function is defined, in our case, in Equation (7). And, by using the k-means approach, the clustering loss can be defined as

$$L_c = \sum_{i=1}^{N} \|\mathbf{x}_i - \mathbf{C}\mathbf{y}_i\|_2^2, \tag{9}$$

where the constant *N* denotes the number of data. $\mathbf{y}_i$ is a *k*-dimensional vector representing the cluster assignment vector of the input $\mathbf{x}_i$. Note that all the elements of $\mathbf{y}_i$ are zero except the element corresponding to the cluster index of the input, whose value is 1. **C** is *M*×*k* matrix, with *M* being the dimension of the input $\mathbf{x}_i$ and *k* represents the number of the clusters. The matrix **C** contains the clusters' centroids that must be learned. Note that the centroids have the same dimension with the input data. Minimizing Equation (9) can be thought of as solving the following problem:

$$\min_{(\mathbf{C} \in \mathcal{R}^{M \times k}),(\mathbf{y}_i \in \mathcal{R}^k)} \sum_{i=1}^{N} \|\mathbf{x}_i - \mathbf{C}\mathbf{y}_i\|_2^2 \text{ such that } y_{j,i} \in \{0,1\}, \mathbf{y}_i{}^T \mathbf{1}_k = 1, \tag{10}$$

where $y_{j,i}$ represents the elements of the assignment vector $\mathbf{y}_i$, with *j* varying from 1 to *k*, and $\mathbf{1}_k$ represents a *k*-dimensional vector containing only the values 1. Note that with $\mathbf{y}_i{}^T \mathbf{1}_k = 1$, the condition that every element of $\mathbf{y}_i$ should be zero except one element, whose value should be 1, is satisfied.

In our situation, the input **x** in Equation (9) and Equation (10) is precisely the latent representations (features) learned by the DCAE, as opposed in [20], where the input **x** represents the output of the CNN. Which means that, at iteration *t* of the training process, the clusters' centroids contained in the matrix **C** and the clusters' assignments are updated according to the latent representations **x** produced by the DCAE at the same iteration.

By minimizing at the same time the reconstruction and the clustering losses, we can expect two important effects for our approach. Firstly, we expect that the DCAE will learn to produce features that are k-means-friendly, which means that the feature space produced by our network is expected to have the property of being easily separated by distinctive clusters. Secondly, the computed centroids at each iteration will be forced to follow the distribution of the features, which reinforces the probability of producing quite distinctive clusters.

The architecture of the proposed DCAE is shown in Table 1. We can distinguishably notice the two parts of the network: the encoder, which reduces gradually the spatial size of the input (see in the 6th column) while increasing the volume's depth (see the 3rd column, the number of feature maps); and the decoder, which gradually increases the spatial size of the input while decreasing the depth. Note that the size of the input image is 112×112.

Table 1. Architecture of the DCAE.

Layer	**Filter size**	**#Feature Maps**	**Stride**	**Padding**	**Output**
Input	-	1	-	-	**112x112**
Conv 1	3x3	32	1	1	112x112
Pool 1	2x2	32	2	0	56x56
Conv 2	3x3	64	1	1	56x56
Pool 2	2x2	64	2	0	28x28
Conv 3	3x3	128	1	1	28x28
Pool 3	2x2	128	2	0	14x14
Conv 4	3x3	256	1	1	14x14
Pool 4	2x2	256	2	0	7x7
Conv 5	7x7	**512**	1	1	**1x1**
Conv 6	7x7	256	1	0	7x7
Unpool 4	2x2	256	2	0	14x14
Conv 7	3x3	128	1	1	14x14
Unpool 3	2x2	128	2	0	28x28
Conv 8	3x3	64	1	1	28x28
Unpool 2	2x2	64	2	0	56x56
Conv 9	3x3	32	1	1	56x56
Unpool 1	2x2	32	2	0	112x112
Conv 10	3x3	1	1	1	**112x112**

One important drawback of the DCAE structure is the down-sampling process performed by the encoder. The input image is systematically down-sampled while we progress inside the network until we arrive at the latent representations' space (or layer), after which the decoding, which means the up-sampling, begins. This has as consequence that the network loses the spatial information of the image layer after layer. By decreasing the size of the image, the local structure of the pixels is also distorted. This distortion complicates the reconstruction process. That is the reason why the preservation of the local structure of the original pixels can be essential. Using the reconstruction loss, as opposed to the fact of using only the clustering loss [16,17], can be the solution to this problem, as discussed by Guo et al. [15]. Instead, we consider that, if incorporating the reconstruction loss in the global loss of the network, as defined in Equation (8), can help in that direction, it is not the only way of assuring a better preservation of the spatial structure of the original pixels.

One of the techniques that can also help to minimize the loss of the cues related to the spatial structure of the data can be the fact of avoiding the use of big filters. As we can see in Table 1, all the convolution operations utilize a 3×3 filter size. Only the last convolutional layer from the encoder (Conv 5) utilizes a different filter size (7×7). Note that the 7×7 filter is used in order to produce the one-dimensional features' layer that follows. Note also that the stride and padding, except for Conv 5, are used in such a way that the output of every convolutional layer has the same spatial extent with its input. That is, the convolution operations do not alter or distort the spatiality of the input. This property also attenuates in a certain level the loss of the information related to the spatiality in the encoder. The down-sampling operation is realized exclusively by the pooling layers. We can see, in Table 1, how every output volume of the pooling layer is spatially down-sampled by half. The layer in the middle of network can be thought of as a one-dimensional vector containing 512 components or elements. This layer, as we can see, contains the features of the DCAE that will be passed to the clustering layer in order to compute the clusters' centroids.

Right after the 1x1x512 feature vector, the up-sampling mechanism in the decoder starts with the stacking of many convolutional and unpooling layers until we reach the original size. Note that we did not use the transposed convolution operations, except for the “Conv 6”, which is the only transposed convolutional layer of the network precisely because it must increase the size of the 1x1x512 feature vector. All the remaining layers, from “Conv 7” to “Conv 10”, apply normal convolution operations. Besides that, every remaining up-sampling process is performed only by the unpooling layers, which are the opposite correspondents of the pooling layers. In Table 1, the unpooling layers are depicted as “Unpool *n*”. Here also in the decoder, the convolution operations do not alter the spatial dimensions of the inputs.

The local structure preservation problem has also been faced in the segmentation topics, where the original image’s spatial information is more than critical [103,104]. We propose here to apply two techniques used in the segmentation’s problems. The first one is to use the position storage technique, as proposed by Badrinarayana et al. [103]. This technique, as illustrated in Fig. 10, consists of storing the positions of the selected activations during the maximum pooling process performed in the encoder. In the decoder, the unpooling process will exclusively consist of placing the activations at the stored positions and setting all the remaining values to zero.

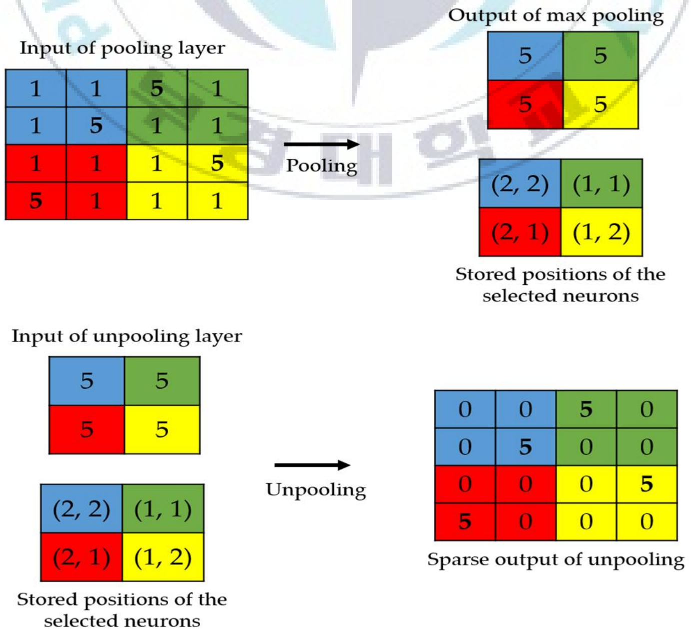


Fig. 10. Illustration of the pooling indices storage technique. The positions of the strongest activations are utilized in order to produce the sparse output.

In Fig. 11, we show the connections between the layers from the encoder and the decoder. The layers shown in red in the encoder are the ones that undergo the maximum pooling operation. While applying the maximum pooling, we store the positions of the strongest activations in the feature map. Note that the strongest activation here refers to the biggest value inside the 2×2 filter of the max pooling, as clearly depicted in Fig. 10. On the other hand, the layers shown in red in the decoder are the outputs of the unpooling operations and they are made by using the stored positions from their corresponding layer in the encoder. All these layers are filled with the sparse-kind outputs depicted in Fig. 10. As we can see in Fig. 11, every output of the unpooling layer goes through a convolution layer whose purpose will be, indeed, to densify the sparse representations. By using this storage technique, the network does not change and the architecture still remains similar with the one depicted in Table 1.

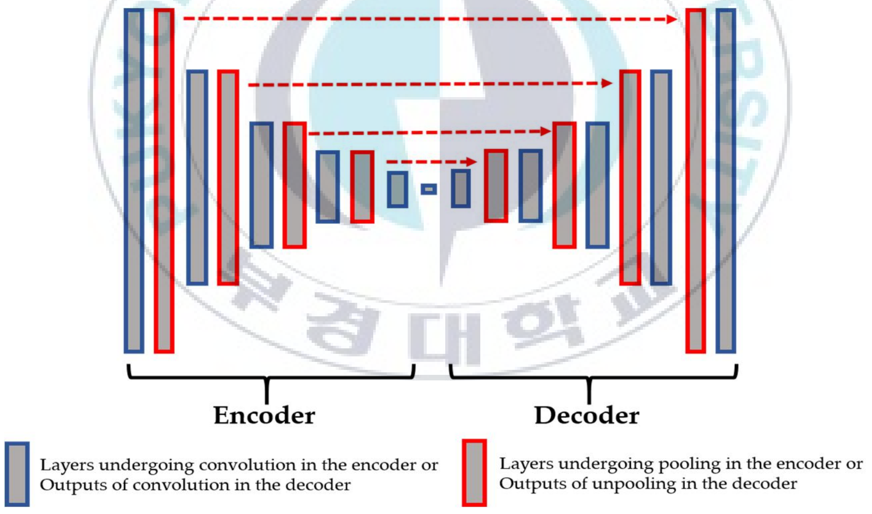

Encoder
Decoder
Layers undergoing convolution in the encoder or Outputs of convolution in the decoder
Layers undergoing pooling in the encoder or Outputs of unpooling in the decoder

Fig. 11. The connections between the pooling layers from the encoder and the unpooling layers from the decoder.

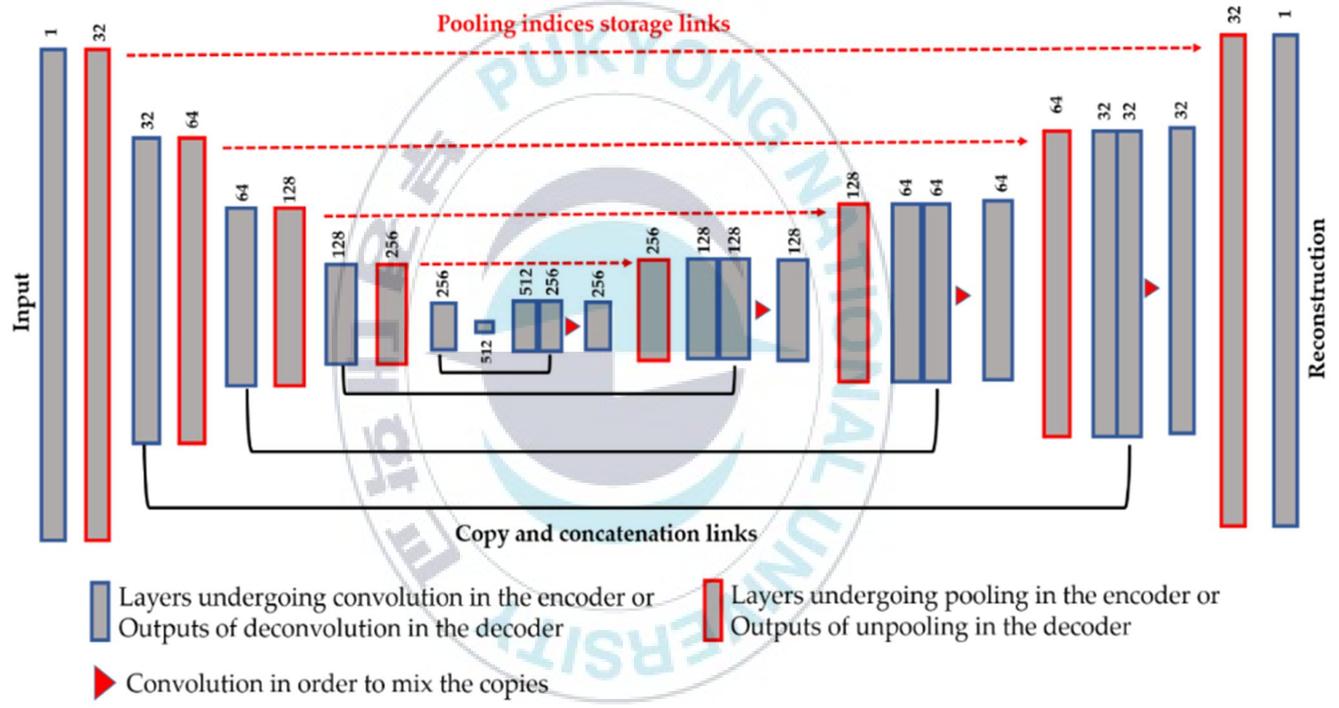

Pooling indices storage links
Input
1
32
32
64
64
128
128
256
256
512
512
256
256
256
128
128
128
128
64
64
64
64
32
32
32
32
1
Reconstruction
Copy and concatenation links
Layers undergoing convolution in the encoder or Outputs of deconvolution in the decoder
Layers undergoing pooling in the encoder or Outputs of unpooling in the decoder
Convolution in order to mix the copies

Fig. 12. Illustration of the proposed network using the storage and copy\concatenate techniques at the same time. The dimensions shown above each layer refers to the number of features maps (the depth of the volume). See Table 1 for the spatial dimensions of each volume.

Another technique in order to assure a better preservation of the local structure is the use of the copy and concatenation operations, as proposed by Ronneberger et al. [104] with the so-called Unet, for the segmentation of the cellular images. This idea consists of mixing, by concatenation operations, the features from the encoder and their corresponding features in the decoder. All of the connections are made before the down-sampling process. Fig. 12 illustrates the copy and concatenate process and provides the details of the network. We propose to use the pooling indices storage and the copy-and-concatenation techniques in the same time (see Fig. 12 for the final structure). The unpooling are done in the same way as described above.

In Fig. 12, we can remark the three main differences with the Unet. First, we did not use any transposed convolution. All the feature extraction process is performed by the convolutional layers and the nonlinearities from the ReLu. Second, the up-sampling operation is exclusively performed by the unpooling layers by using the pooling indices storage technique. Third, as expected, the final layer of our network comprises one single channel and represents the reconstruction of the original images, and not the segmentation mask. As we can see in Fig. 12, after every copy-and-concatenation process, we apply a convolution operation, represented by the red triangle in Fig. 12, whose purpose is precisely to mix up the information from the encoder and the decoder. When the copy-and-concatenation process increases the depth of a volume, the following convolution will combine the information and permit to get back to the original volume's depth. The network shown in Fig. 12 has slightly the same structure than the one in Table 2, the only difference being the copy-and-concatenation layers. In chapter 4, we will discuss how all these techniques can affect the quality of the features and, thus, the quality of the learned clusters.

3.1.3 Deep embedded clustering with dual autoencoder

In this section, we present a modified version of the DCAE with an embedded clustering layer. We propose to use a dual autoencoder that performs a double feature extraction and clustering. Fig. 13 shows the illustration of the deep embedded dual autoencoder.

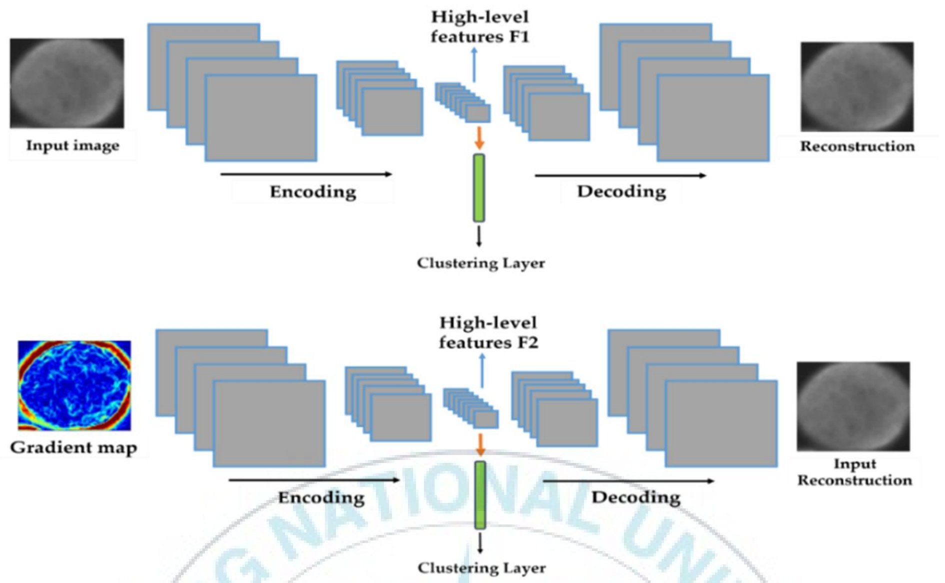


Fig. 13. Illustration of the proposed dual embedded network.

This idea is slightly inspired by the dual DCAE proposed in [21] and explained in section 2.2. In our version, we propose to use two levels of feature extraction with the DCAE. The first network will take as input the original cellular image and will learn to reconstruct it by using the decoding function depicted in Equation (5). The original image contains the intensity and geometric information concerning the cells. Our assumption is that the high-level features learned by this network will encapsulate the intensity and geometric information about the cellular patterns.

The second network will take the gradient magnitude of the image as the input. In every single pixel of the image, the gradients $\overrightarrow{\nabla \mathbf{I}}$, evaluated using the following equation

$$\overrightarrow{\nabla \mathbf{I}} = \frac{\partial \mathbf{I}}{\partial x}\mathbf{e}_x + \frac{\partial \mathbf{I}}{\partial y}\mathbf{e}_y, \tag{11}$$

compute the rate and the direction of the changes in the intensity variation. In Equation (11), $\mathbf{I}$ represents the original image and the unit vectors $\mathbf{e}_x$ and $\mathbf{e}_y$ represent the two axis of the image, the horizontal and vertical directions along which we compute the changes in pixel level. The gradient magnitude is the magnitude of the vector $\overrightarrow{\nabla \mathbf{I}}$, whose estimation, following Equation (11), can be written as

$$\mathbf{G(I)} = \sqrt{\left(\frac{\partial \mathbf{I}}{\partial x}\right)^2 + \left(\frac{\partial \mathbf{I}}{\partial y}\right)^2} \tag{12}$$

where **G(I)** matrix represents the gradient magnitude of the image **I**.

While the encoder of the first network uses the expression denoted in Equation (4) in order to compute its output, in the second network, we replace the input $\mathbf{x}$ by its gradient magnitude. Hence, the output of the encoder from the second DCAE can be re-written as

$$\mathbf{y} = g(\boldsymbol{\theta} \cdot \mathbf{G}(\mathbf{x})), \tag{13}$$

where $\boldsymbol{\theta}$, again, englobes all the different parameters of the encoder. Note that the reconstruction process of the second DCAE is done in the same manner as the one of the first network. Equation (5) is used for computing the output of the decoder, and Equations (6) and (7) are used in the same manner in order to find the best parameters that minimize the most the differences between the decoder's output and the original cellular image. Which means that the second network takes as inputs the gradients and, using them, try to reconstruct the original cell image.

The second assumption made here is that the gradient maps will allow the second network to seize and understand the local changes in intensity level of the cellular images. And, as previously mentioned, the first network that takes the original images will seize the intensity and geometric information of the cellular images. The architecture of both networks is also depicted in details in Table 2, which was discussed in details in the previous section. It is quite important to note here that the method proposed in this section is just an indication of how can efficiently utilize the concept of dual autoencoder. Results using this method are not shown in this work since it is not our main proposed method. Readers who are interested in this technique are encouraged to refer to [3].

3.1.4 Attention-based deep clustering

We propose another architecture in order to boost the reconstruction consistency of the network. In the previous sections, we discussed how the pooling indices storage and the copy-and-concatenation techniques can help to ensure a better reconstruction for the decoder. In this part, we continue in that way by proposing a system of three reconstructions performed in parallel. This idea comes from the attention networks proposed by Wang and Shen [105] where three streams are utilized in order to generate the attention map. Attention maps predict, in an image, the locations where human's eyes focus the most. Details about this attention prediction can be found in [105].

Note that in the attention generation methods, the final "attention maps" are computed in a supervised learning way, similarly as done in the generation of the segmentation mask [103,104]. Cross-entropy is used for the purpose of minimizing the differences between the network's output and the labels denoting the "attention map". In this work, the attention-based network is utilized in an unsupervised way, just like for a normal DCAE.

Fig. 14 and Fig. 15 demonstrate the idea of attention-based deep clustering. We can clearly distinguish the three different streams in the two figures. The encoder has one stream. The idea is to apply an up-sampling process before the down-sampling goes further in the encoder. We already discussed previously about the disadvantages of the down-sampling process. An intuitive understanding of the proposed attention-based network is the minimization of the down-sampling's effects by using the encoder's volumes for the decoding (up-sampling) process before they lose their spatial dimensions. In other words, this technique consists of launching the reconstruction process right before the feature maps are down-sampled.

Layer	Filter size	#Feature Maps	Output Stream 1	Output Stream 2	Output Stream 3
Input	-	1	112 × 112	112 × 112	112 × 112
Conv 1	3 × 3	32	112 × 112	112 × 112	112 × 112
Pool 1	2 × 2	32	56 × 56	56 × 56	56 × 56
Conv 2	3 × 3	64	56 × 56	56 × 56	56 × 56
Pool 2	2 × 2	64	28 × 28	28 × 28	28 × 28
Conv 3	3 × 3	128	28 × 28	28 × 28	28 × 28
Pool 3	2 × 2	128	14 × 14		14 × 14
Conv 4	3 × 3	256	14 × 14		14 × 14
Pool 4	2 × 2	256	7 × 7		
Conv 5	7 × 7	**512**	**1 × 1**		
Deconv 6	7 × 7	256	7 × 7		
Unpool 4	2 × 2	256	14 × 14		
Conv 7	3 × 3	128	14 × 14		
Unpool 3	2 × 2	128	28 × 28		28 × 28
Conv 8	3 × 3	64	28 × 28		-
Unpool 2	2 × 2	64	56 × 56	56 × 56	56 × 56
Conv 9	3 × 3	32	56 × 56	-	-
Unpool 1	2 × 2	32	112 × 112	112 × 112	112 × 112
Conv 10	3 × 3	1	112 × 112	112 × 112	112 × 112
Fusion	-	**3**		112 × 112	
Conv-Fusion	1 × 1	1		**112 × 112**	

Fig. 14. Architecture of the attention-based network.

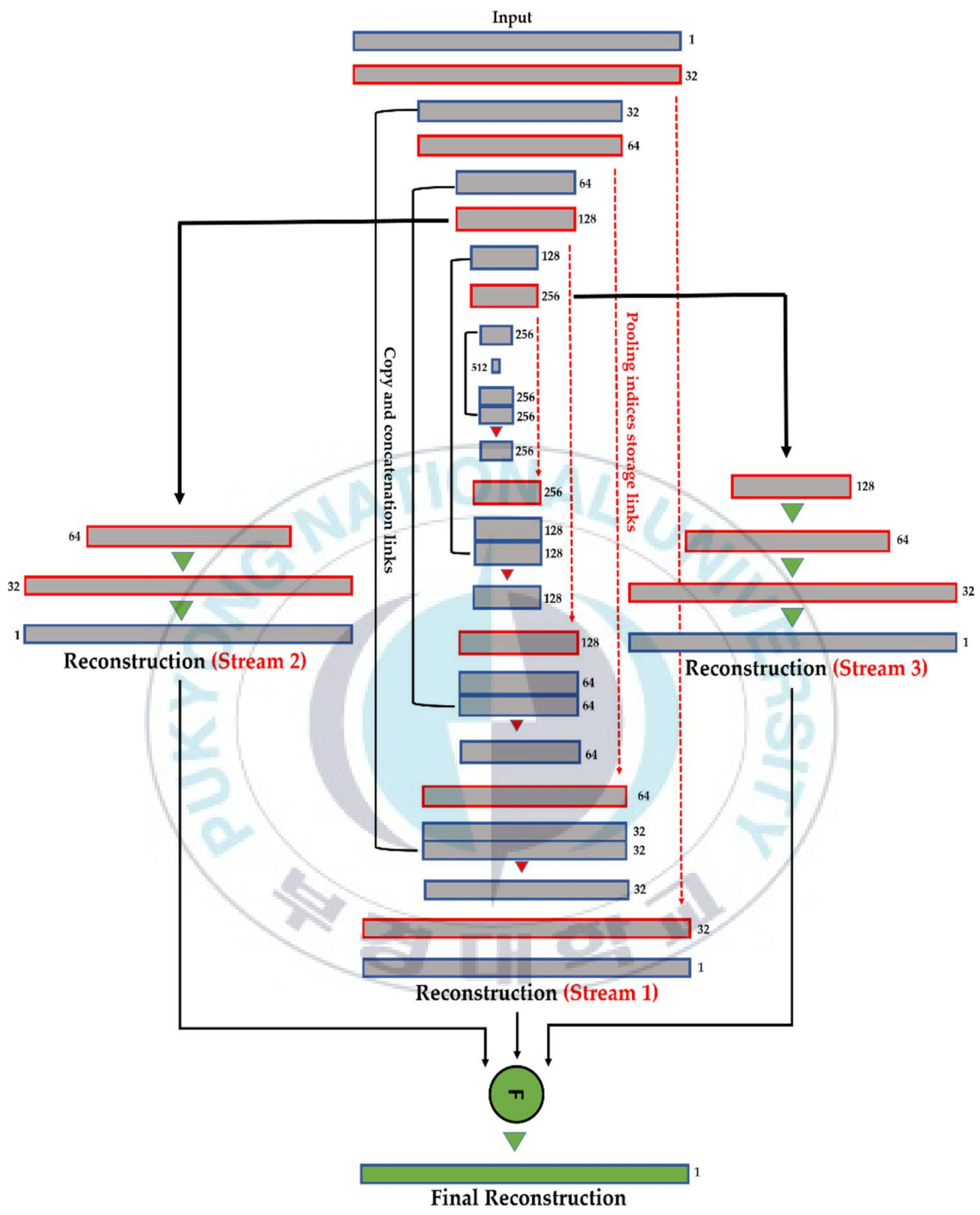

Input
1
32
32
64
64
128
128
256
256
512
256
256
256
256
128
128
128
128
64
64
64
64
32
32
32
32
1
Copy and concatenation links
Pooling indices storage links
64
32
1
Reconstruction (Stream 2)
128
64
32
1
Reconstruction (Stream 3)
Reconstruction (Stream 1)
F
1
Final Reconstruction

Fig. 15. Illustration of the proposed deep attention-based network.

The architecture of the attention network is similar with the one used for the previous cases. This architecture is explained in details in the previous sections. Here, we can just emphasize on the fact that the reconstructions from the three different streams are mixed together in order to produce the final reconstruction. This mixing process is firstly conducted with a concatenation process. The layer named "**Fusion**" in Fig. 14 performs the concatenation of the initial reconstructions accomplished by the three streams. The reconstruction maps produced by the three different streams are concatenated and passed through a 1x1 convolution (see the layer "**Conv-Fusion**" in Fig. 14) whose purpose is precisely to mix up the information from the three streams. The final output will represent the final reconstruction of the network. Fig. 15 also gives an illustration about this fusion process. The fusion is illustrated by the green circle with the letter "F" in Fig. 15.

Note that in this attention-based network, the pooling storage and copy-and-concatenation procedures are accomplished only in the first stream, the main stream. In the second and third streams, the pooling indices storage is not performed because the up-sampling process is accomplished only by the transposed convolution operations. And the copy-and-concatenation shortcuts are avoided in the second and third streams. Note that they can also be performed but we could not see any effective improvement in the reconstruction by adding these two procedures while they add a huge complexity in the network. We recall that, with all of these processes, a clustering layer is still embedded in the middle of the network (here, in the first stream) in order to cluster the features produced by the DCAE. We conduct a systematic analysis in order to see at which point these techniques can improve the quality of the reconstruction. At the same time, we will try to analyze if the reconstruction quality provided by these techniques will, in fine, improve the quality of the learned features, thus, improving the quality of the final clustering process.

3.2 Proposed active learning method

3.2.1 Method overview

In this part, we present the semi-supervised learning approach that we used. The semi-supervised learning proposed here is based on active learning. In the previous chapter (see section 2.4), we presented in details the idea behind active learning. Since this kind of method is based on the supervised approach, we start by presenting the networks that we have used in that purpose. We have proposed the dynamic networks for the cellular images, since we have used these images as the principal data for the experiments.

Thus, in the first part of this section, we present in details the reason why we have used the dynamic networks. We present the justification beyond this choice according to the nature of the data that we have. In the first paragraphs of the next section, we present the characteristics of the HEp-2 cell datasets. After that, we present the networks that were designed specifically in order to tackle the problems faced in the HEp-2 cell classification literature.

After this step, we present the first part of the active learning scheme using the proposed networks. This first active learning scheme is based on the conventional way of selection. In the last section, we introduce our newly developed active learning scheme that uses the XAI for the selection process.

3.2.2 Deep parallel residual networks

The first step of our method is to create the parallel networks that we will use for the transfer and active learning. HEp-2 cell datasets have the particularity of denoting a significant heterogeneity. This is caused by the existence of mainly two different levels of fluorescence illumination (also denoted as intensity levels). Images shown in Fig. 16 illustrate the disparities caused by the inhomogeneous fluorescence illumination. These disparities are the intra-class variations and the inter-class similarities. Intra-class variations denote the variations within the same cellular type. Fig. 16 (a) shows a randomly selected positive intensity Nucleolar image, while Fig. 16 (b) shows a randomly selected negative intensity Nucleolar image. We can remark the strong disparities in terms of intensity between the two images even though they belong to the same class (intra-class variations). The same dissimilarities can be noticed between the two images depicted in Fig. 16 (c) and Fig. 16 (d), in case of the Nuclear membrane cell type.

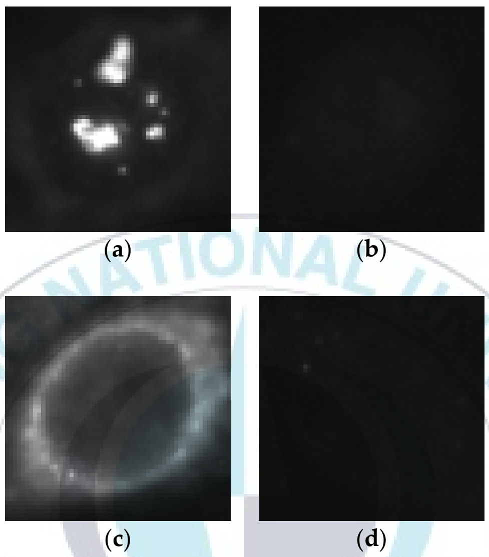


Fig. 16. HEp-2 cellular images from the 13A dataset. (**a**) A positive intensity Nucleolar cell; (**b**) A negative intensity Nucleolar cell; (**c**) A positive intensity Nuclear membrane cell; (**d**) A negative intensity Nuclear membrane cell.

Inter-class similarities, on the other hand, denote the similarities that exist between the different classes. In fact, the images shown in Fig. 16 (b) and Fig. 16 (d) exhibit strong similarities in terms of intensity even though they belong to two different cellular types. This heterogeneity-related problem really adds complexity in the HEp-2 cell classification task. As a matter of fact, different methods have been proposed in order to specifically classify the different fluorescence intensity [106,107]. Furtherly, Nigam et al. [108] have proposed to perform an intensity-based classification prior to the cell classification itself in order to alleviate the heterogeneity during the cell type classification. Our proposed parallel deep residual networks try to tackle this heterogeneity-related problem in one step (unlike in [108]) and by performing cell type classification (unlike in [106] and [107]).

We propose to use the different wavelet coefficients from the 2D-DWT decomposition as the inputs of different networks in parallel. This idea was fully discussed and its effectiveness demonstrated in [109]. We have upgraded the idea by alleviating the learning (training) process by reducing the total number of needed networks and, consequently, the total number of the parameters to handle. The 2D-DWT in the first level produces 4 different matrices of coefficients. The approximation coefficients, which represent the low-frequency information of the inputs, and the three different details coefficients, which represent the high frequency components of the input signal. The three details coefficients are the horizontal, vertical and diagonal details. Unlike in [109], where four different networks were utilized in parallel for all the four coefficients, we sum up all the details coefficients in order to incorporate all the high frequency components in one single channel. Thus, as illustrated in Fig. 17, two networks are trained in parallel: the first network takes the approximation coefficients as the inputs, and the second network takes the sum of all the three details coefficients as inputs.

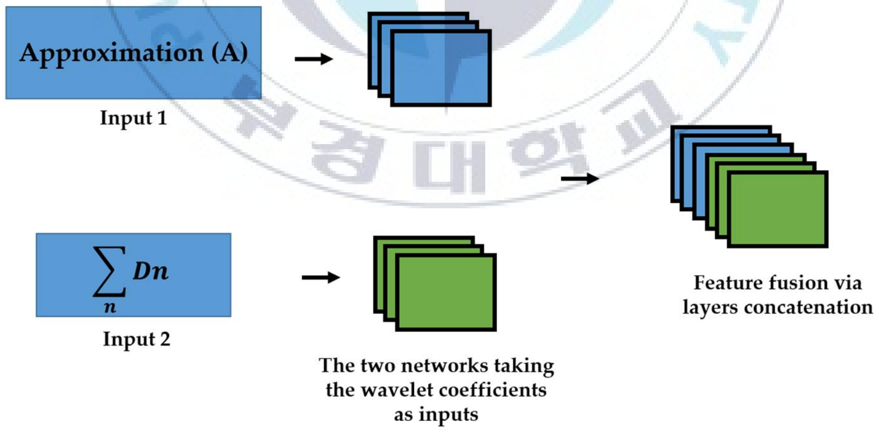

Approximation (A)
Input 1
$\sum_n Dn$
Input 2
The two networks taking the wavelet coefficients as inputs
Feature fusion via layers concatenation

Fig. 17. Illustration of the proposed parallel networks: The approximation (A) and the sum of details, represented by *D*, are given to two networks. A feature fusion is performed in the late layers. Note that *n* here is 3 and represents the three details components.

The approximation coefficients will bring a certain homogenization in terms of the intensity. This will drastically reduce the intra-class variations by forcing both the positive and negative intensity images to share a similar level of gray intensity. Images shown in Fig. 18 illustrate the intensity-based homogenization produced by the approximation coefficients. Fig. 18 (a) shows a randomly selected positive intensity Fine speckled cell image from the SNPHEp-2 dataset. Fig. 18 (b) shows its corresponding approximation coefficients (extracted from the first level of the 2D-DWT decomposition). Fig. 18 (c) shows a randomly selected negative intensity Fine speckled cell image and Fig. 18 (d) shows its corresponding approximation coefficients. We can clearly remark the homogenization that occurred between the images in terms of the intensity of the gray level by comparing their approximation coefficients. This homogenization will drastically reduce the intra-class variations of the dataset.

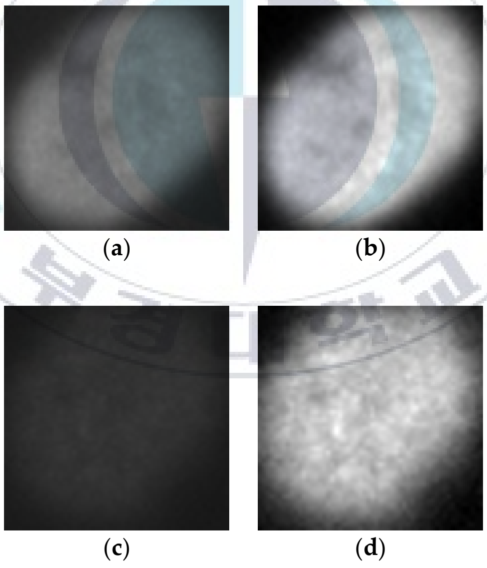


Fig. 18. Approximation coefficients of the HEp-2 cellular images from the SNPHEp-2 dataset. (**a**) A positive intensity Fine speckled; (**b**) Its approximation coefficients; (**c**) A negative intensity Fine speckled; (**d**) Its approximation coefficients. Note the effective homogenization in terms of gray level intensity between the positive (**b**) and negative (**d**) images.

Secondly, the details coefficients will bring homogenization in terms of the geometrical shape of the cells. In fact, the details coefficients capture the high frequency components of the image, which means that all the gray variations inside the image can be highlighted. These small changes in intensity indicate the shape and the boundaries of the cells. Images depicted in Fig. 19 illustrate how the high-frequency components can help to expose the cellular shape and boundaries from the positive and negative intensity images.

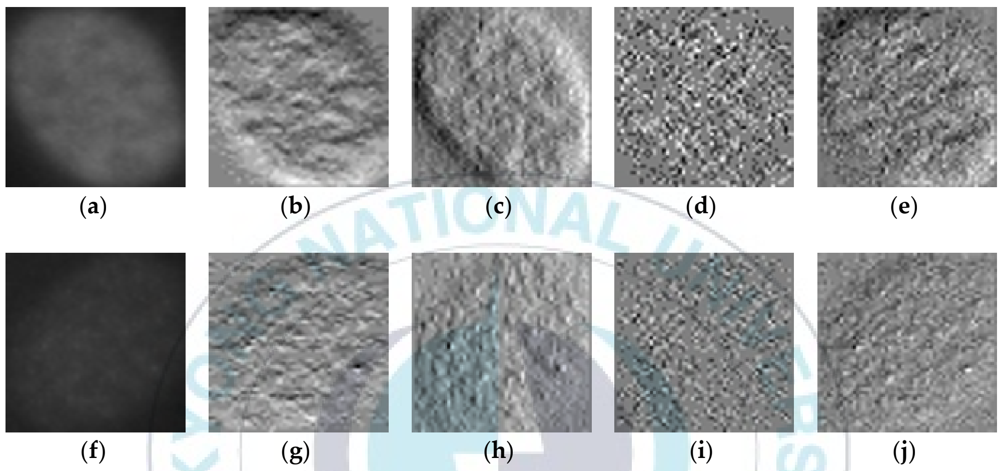
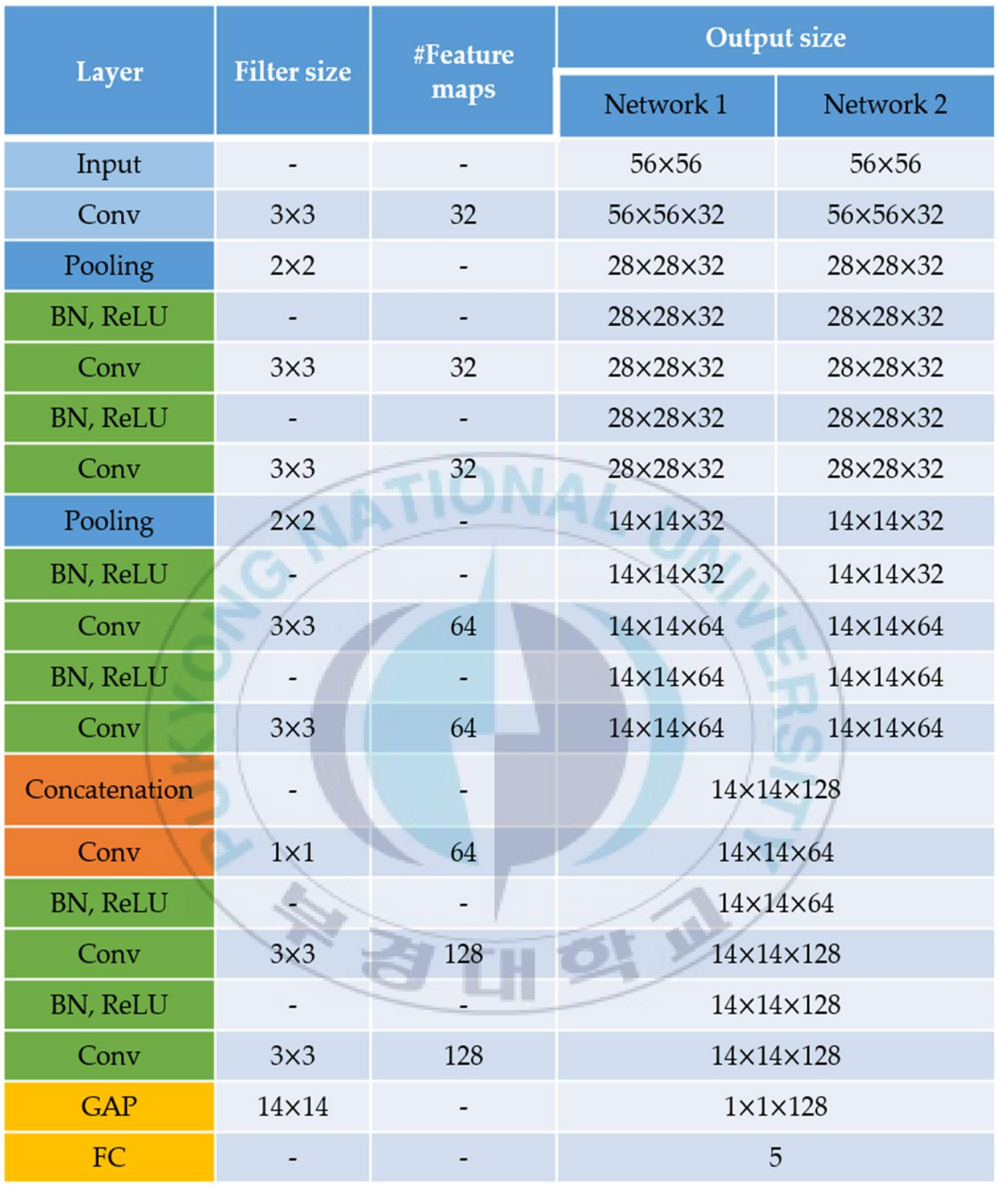


Fig. 19. Details coefficients of the HEp-2 cellular images from the SNPHEp-2 dataset. (**a**) A positive intensity Homogeneous; (**b**)-(**e**) its horizontal, vertical, diagonal details and their sum, respectively. (**f**) A negative intensity Homogeneous; (**g**)-(**j**) its horizontal, vertical, diagonal details and their sum, respectively. The original images in (**a**) and (**f**) have a size of 112 × 112. Their respective detail coefficients in (**b**)-(**e**) and (**g**)-(**j**) are all downsized by half (56 × 56). All the images in this figure were identically resized for the purpose of visualization.

In Fig. 19 (a), we have a positive intensity Homogeneous cell image. Images shown in Fig. 19 (b)-(d) represent its different details coefficients, the horizontal, vertical and diagonal details, respectively. The image shown in Fig. 19 (e) is the result of summing all the details. We can remark that the sum incorporates all the information from the three details coefficients. Similarly, in Fig. 19 (f), we show a negative intensity Homogeneous cell image. In Fig. 19 (g)-(i), we have the three details coefficients and Fig. 19 (j) represents their sum. Note how the two sums (Fig. 19 (e) and Fig. 19 (j)) highlight the Homogeneous cell's shape, boundaries and internal gray variations. Because these three elements differ from a cellular type to another, we can expect two main contributions from the sum of details. First, they will bring a certain heterogeneity between the classes by forcing all the negative intensity images to exhibit typical characteristics of their cellular type (shape, boundaries and gray variations). This will contribute to the reduction of the inter-class similarities.

Consequently, as the second contribution, they will bring a certain intra-class homogenization by forcing the positive and negative intensity images from the same class to exhibit similar patterns (shape, boundaries and gray variations), as demonstrated by Fig. 19 (e) and Fig. 19 (j). This will contribute to the reduction of the intra-class dissimilarities, reduction also achieved by the approximation coefficients, as previously discussed. The approximation and the sum of details will be used to feed two residual networks in parallel.

Fig. 20 shows the architecture of the residual networks. Network 1 takes the approximation coefficients while Network 2 takes the sum of the details, as explained above. There are five residual blocks in total. Each network has two residual blocks and another is used after the feature fusion from the two networks. Every residual block has two convolutional layers, two rectified linear unit (ReLU) layers, and two batch normalization layers.



Layer	Filter size	#Feature maps	Output size	
			Network 1	Network 2
Input	-	-	56×56	56×56
Conv	3×3	32	56×56×32	56×56×32
Pooling	2×2	-	28×28×32	28×28×32
BN, ReLU	-	-	28×28×32	28×28×32
Conv	3×3	32	28×28×32	28×28×32
BN, ReLU	-	-	28×28×32	28×28×32
Conv	3×3	32	28×28×32	28×28×32
Pooling	2×2	-	14×14×32	14×14×32
BN, ReLU	-	-	14×14×32	14×14×32
Conv	3×3	64	14×14×64	14×14×64
BN, ReLU	-	-	14×14×64	14×14×64
Conv	3×3	64	14×14×64	14×14×64
Concatenation	-	-	14×14×128	
Conv	1×1	64	14×14×64	
BN, ReLU	-	-	14×14×64	
Conv	3×3	128	14×14×128	
BN, ReLU	-	-	14×14×128	
Conv	3×3	128	14×14×128	
GAP	14×14	-	1×1×128	
FC	-	-	5	

Fig. 20. Architecture of the residual networks. The residual blocks are shown in green. Here, "Conv", "BN", "ReLU", "GAP", and "FC" denote, respectively, the convolutional layer, the batch normalization layer, the rectified linear unit (ReLU) layer, the global average pooling layer, and the fully connected layer. "Pooling" denotes the maximum pooling layer and "Concatenation" denotes the layer that performs feature concatenation from the two networks.

Two main observations need to be made about the architecture in Fig. 20. First, all the convolutional layers preserve the spatial dimensions of the input volume and only the pooling layers perform the spatial down-sampling of the input. Second, the feature fusion is performed by the 1×1 convolutional operation that directly follows the concatenation. After concatenating the layers from the two networks, we obtain a volume of size 14×14×128, which is then passed through the 1×1 convolutional layer whose purpose is precisely to mix (fusion) the information from the two networks. The output volume of the final residual block has the dimensions of 14×14×128. This volume is given to the global average pooling (GAP) layer in order to obtain the final one-dimensional feature vector of size 1×1×128.

The feature vector will be given to a softmax classifier that uses the function defined as follows:

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{i=1}^{N} e^{z_i}}, \quad \text{for } j = 1, \dots, N, \tag{14}$$

where N is the number of the classes and the values z are the inputs of the softmax function. The values $\sigma(z)_j$ are the outputs of the function and represent the probabilities of every class. The parallel networks learn by back-propagating the error [110] and using the cross-entropy error function defined by

$$E = -\sum_{j=1}^{N} y_j \log\left[\sigma(z)_j\right], \tag{15}$$

where the values y_j denote the actual labels of the N classes for a given data and the values $\sigma(z)_j$ are the ones computed using Equation (14).

These parallel networks will be first trained with a small dataset, which contains only around one thousand training instances. After this initial training process, the networks will be utilized as the pre-trained model in order to perform transfer learning coupled with active learning on the targeted dataset, which contains more than sixty thousand instances. Instead of using networks that were pre-trained on ImageNet, as done by most of the works that utilize transfer learning [94,96-98], we propose to use our own networks, which are pre-trained purposely by using the HEp-2 images.

The advantage is that using a network that has previously seen similar images during transfer learning allows to alleviate the parameters' update. Because the two datasets share the similar image domains, we expect them to share many general characteristics. The early layers from the networks, which learn low-level and non-specific features, can be fixed during the fine-tuning. And only the late layers, which learn domain-specific features, can be updated (see Fig. 21 for the illustration). This will smooth and ease the first step of our active learning scheme consisting of fine-tuning the pre-trained model on a very small number of selected data.

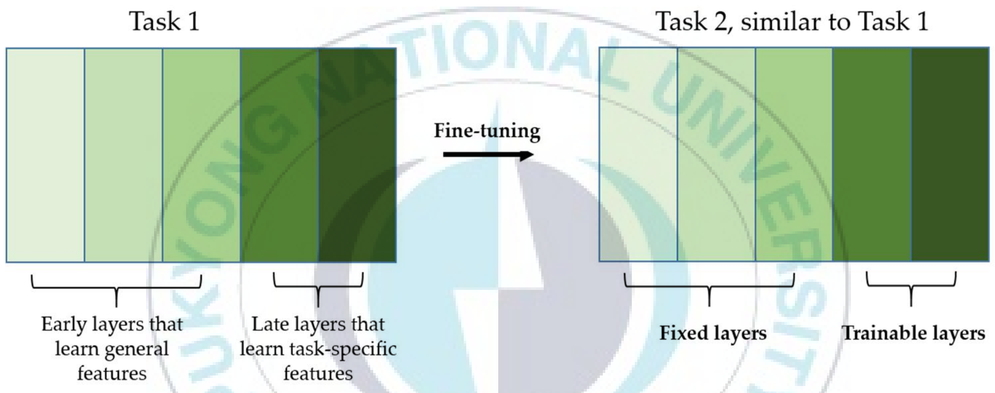

Task 1
Task 2, similar to Task 1
Fine-tuning
Early layers that learn general features
Late layers that learn task-specific features
Fixed layers
Trainable layers

Fig. 21. Illustration of fine-tuning. Only the late layers will be trainable during the fine-tuning. The different shades of green indicate how the specificity of the features increases layer after layer. From left to right, we have the general features to the task-specific features.

3.2.3 Active learning using the pre-trained parallel residual networks

For clarity, we summarize the different steps of the proposed active learning scheme in Table 2. Fig. 22 illustrates these different steps. This process is repeated as much as possible and stopped until we reach our limitations in terms of labeling. As we can remark, the process can be continued until the totality of the data are labeled. But, in our work, we impose to ourselves a limit in terms of the possibility of labeling. In fact, the goal of this work is to demonstrate that active learning-based labeling can really help to limit the need of labeled data while maintaining a fair performance. In our experiments, we explore different hypothesizes concerning the limitations that we have in terms of labeling. For example, if we suppose that we can only label 10% of the 64,000 available data, we stop the process when we reach 6,400 annotated data and evaluate the networks over the testing data. All the details concerning the parameters of the networks, the values *k* and *m*, and the datasets are discussed in the next section.

Table 2. The different steps of selection using the conventional active learning scheme.

Step no.	Actions	Comments
1	We train our networks using the small dataset.	The small dataset used here is the SNPHEp-2 dataset which contains around 1000 images for training.
2	Using the targeted dataset, we select randomly and label k samples. We fine-tune the networks by using these k samples as the training data. As described in Section 3.2.2, the early layers remain fixed and we only update the late layers.	Note that by choosing the number k, we select randomly (as opposed to select by using active learning) the data to label. In fact, we want this number k to be as small as possible, in order to not complicate the labeling process. This is made possible by the pre-training made in step 1 using the small dataset.
3	We use the fine-tuned networks over all the remaining data in order to get their probability scores. We compute the confidence (entropy) using Equation (1) for each data.	Equation (2) and Equation (3) can also be used to estimate the confidence.
4	We rank the data according to their confidence, from the lowest to the highest.	Note that in Fig. 22, we show the data with the lowest confidence (highest entropy) in the bottom for the illustration purpose.
5	We select the first m data in the ranking in step 4 and annotate them. These are the data for which the networks are the most uncertain about.	The number m is chosen according to the limitations that we have in terms of manual labeling.
6	The newly annotated data in step 5 are mixed with the k data that were previously labeled in step 2 in order to create the newly annotated set.	The newly annotated dataset contains now $k + m$ data.
7	We fine-tune again the networks using this newly annotated dataset	After this step, we get back to step 3 (use the newly fine-tuned networks to compute the scores and the confidence).

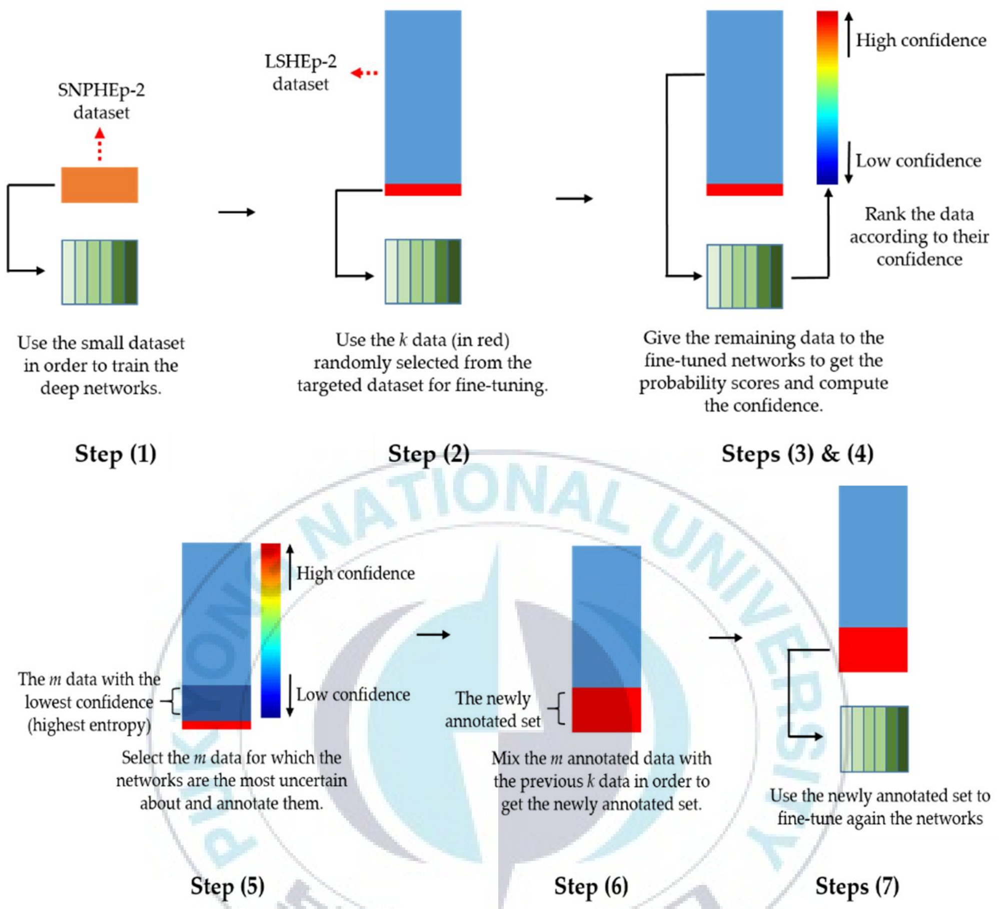

SNPHEp-2 dataset
Use the small dataset in order to train the deep networks.
Step (1)
LSHEp-2 dataset
Use the k data (in red) randomly selected from the targeted dataset for fine-tuning.
Step (2)
High confidence
Low confidence
Rank the data according to their confidence
Give the remaining data to the fine-tuned networks to get the probability scores and compute the confidence.
Steps (3) & (4)
The m data with the lowest confidence (highest entropy)
High confidence
Low confidence
Select the m data for which the networks are the most uncertain about and annotate them.
Step (5)
The newly annotated set
Mix the m annotated data with the previous k data in order to get the newly annotated set.
Step (6)
Use the newly annotated set to fine-tune again the networks
Steps (7)

Fig. 22. Illustration of the different steps of the proposed active learning scheme. LSHEp-2 stands for the large-scale HEp-2 dataset.

3.2.4 Explainable active learning

In this part, we present a novel active learning scheme based on XAI. The core idea in this scheme is to use the relevance maps produced by the XAI algorithm as the focus of the analysis. As developed in section 2.4, conventional active learning methods utilize the direct outputs of the network, which are the probabilities associated with each one of the classes, as the reference for measuring or estimating the uncertainty of the model.

Also, as discussed previously, deep learning models are qualified as “black-box” in the sense that it is very difficult to interpret the decisions made by the model. By using the model’s output, we only transfer the non-interpretability of the model to the selection procedure. Because the model’s output is, by itself, non-interpretable, any selection process based on the model’s output is also, by definition, non-interpretable. The main and principal reason of our choice of utilizing the relevance maps is to make the selection process (or decision) as interpretable as possible, which means, we want the oracle to understand why a data and not another is selected in order to be annotated.

The second and also very important reason is that this selection process is completely task-agnostic. Selection methods that are based on the model’s output are task-dependent in the sense that, because we are using the model’s output, the selection process also depends on the nature of that output. In practice, a selection method that utilizes the output cannot be used, in the same manner, to models that are being utilized for different tasks. For example, the outputs of classification and regression problems are fundamentally different. Thus, it is inconsistent to use in the same manner a selection method that is based on the model’s output for both classification and regression. The selection should be adapted for each one of them.

On the contrary, a selection process that uses only the relevance maps does not depend on the nature of the model’s outputs. The relevance maps are constructed using the inner reactions of the models, which means that the XAI algorithm literally maps the regions (or locations) of the networks that are most excited for a given input in order to produce a given output. In other words, we aim to transfer the selection problem from the “network’s output space”, which necessarily depends on the nature of the task, to the “relevance map’s space”, which completely ignores the nature of the task.

The third reason that justifies the use of relevance maps is that they tend to not really be correlated with the model's performance. We admit that, at some point, they can depend on the discriminability of the data. Furthermore, the most important assumption made here in order to motivate the choice of our proposed type of selection is that the original data possess some internal characteristics that make them to be unique. If we suppose, at the first point, that an effective discrimination is possible on some data, then, this proposed type of selection process is also justified.

Why? Because the relevance maps are precisely the elements that reveal, expose and highlight the regions, in the original data, that made the discrimination to be possible in the first place. Relevance maps are the best ways to precisely focus only on those relevant regions that characterize the data in their category. Thus, any selection method that uses relevance maps will be at least as efficient as the selection based on the output. With the advantage of being interpretable. The illustration of the proposed explainable active learning scheme is portrayed in Fig. 23. We made a succinct description of every single step of the proposed explainable active learning in Table 3.

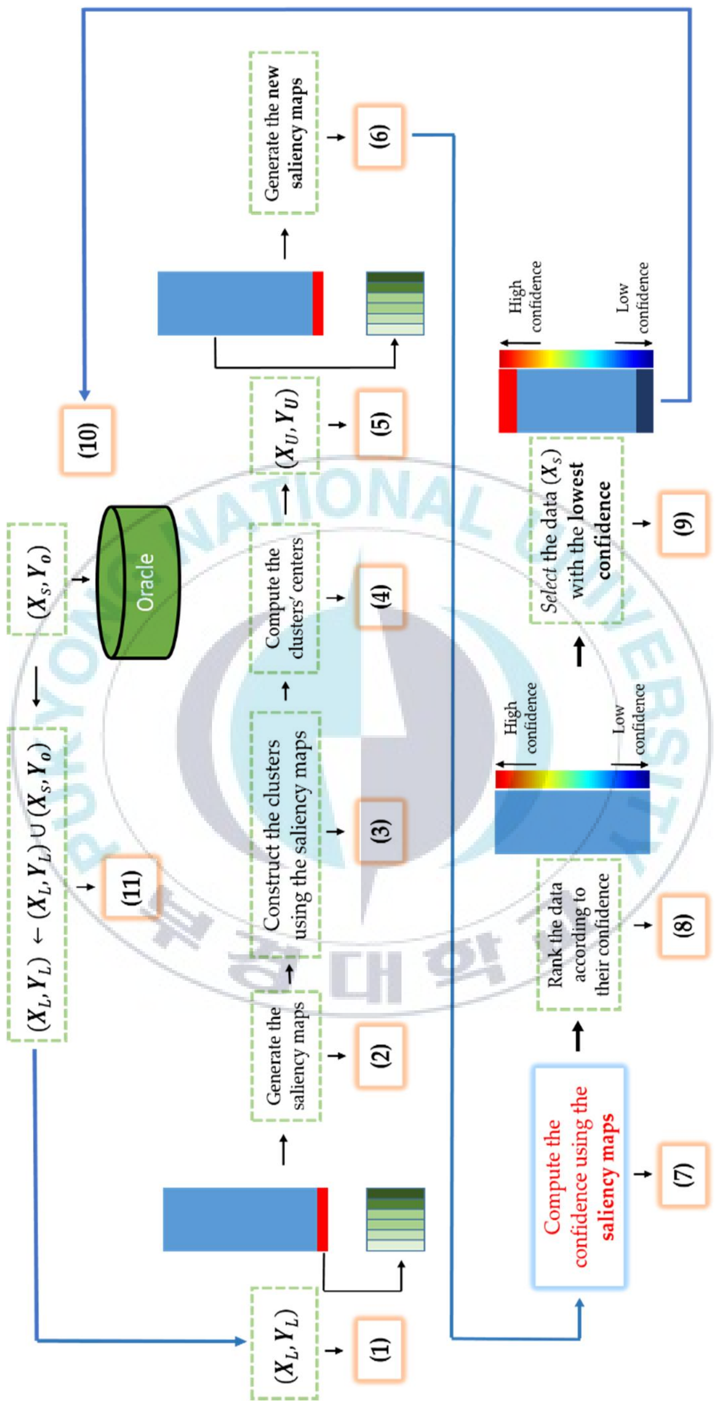

(X_L, Y_L)
(1)
Generate the saliency maps
(2)
Construct the clusters using the saliency maps
(3)
Compute the clusters' centers
(4)
(X_U, Y_U)
(5)
Generate the new saliency maps
(6)
Compute the confidence using the saliency maps
(7)
Rank the data according to their confidence
(8)
High confidence
Low confidence
Select the data (X_S) with the lowest confidence
(9)
High confidence
Low confidence
(10)
(X_S, Y_O)
Oracle
$(X_L, Y_L) \leftarrow (X_L, Y_L) \cup (X_S, Y_O)$
(11)

Fig. 23. Illustration of the different steps of the proposed explainable active learning.

Table 3. The different steps of the proposed explainable active learning.

Step no.	Actions	Comments
1	Create the initially labeled data ($\boldsymbol{X}_L$, $\boldsymbol{Y}_L$)	The selection is done randomly for this initial step.
2	Generate the relevance maps of the data using LRP ($\epsilon = 100$) or DeepTaylor maps. Every data X_i is then represented by its relevance map: • $X_i \to \mathcal{F}_{rel}(X_i)$: as an image or • $X_i \to \{\mathcal{F}_{rel}(X_i)\}$: as a vector containing the *N* most relevant values	In this experiment, we represented the data as the whole map. Note that, in case there is a need of reducing the complexity of the clustering process, the second representation can be adopted.
3	• Construct the clusters using the maps $\mathcal{F}_{rel}(X_i)$ as the inputs ▪ Using k-means with $L_c = \sum_{j=1}^{k} \sum_{i=1}^{N} d[\mathcal{F}_{rel}(X_i),\, C_k]$ ▪ Using Agglomerative clustering with average linkage where the clusters are merged with ✓ $\frac{1}{\lvert C_1 \rvert \cdot \lvert C_2 \rvert} \sum_{x \in C_1} \sum_{y \in C_2} d(x, y)$. ✓ Note that x and y are $\mathcal{F}_{rel}(X_i)$	We used agglomerative clustering with the average linkage as the principal clustering method.
4	Compute the clusters' centers: $$C_k = \underset{y \in \{K\}}{\operatorname{argmin}} \sum_{i}^{\lvert K \rvert} d(y,\, x_i)$$	The medoids are used as the clusters' centers instead of the means.
5	Give the unlabeled set ($\boldsymbol{X}_U$) to the network	-
6	Step 2 is repeated for the unlabeled set.	-

7	For every data X_i in the unlabeled set, compute the **confidence**.	The confidence is redefined by using the distance between a given map and the clusters' centers computed in step 4.
8	Rank the data according to their computed confidence.	-
9	Select the data ($\boldsymbol{X_s}$) with the lowest confidence possible	-
10	Annotate the selected data to create the pairs ($\boldsymbol{X_s}$, $\boldsymbol{Y_o}$)	$\boldsymbol{Y_o}$ represents the labels assigned by the oracle.
11	Mix the initially labeled data $(\boldsymbol{X_L}, \boldsymbol{Y_L})_{\boldsymbol{ini}}$ from step (1) with the newly annotated set ($\boldsymbol{X_s}$, $\boldsymbol{Y_o}$)	$(\boldsymbol{X_L}, \boldsymbol{Y_L})_{\boldsymbol{new}} = (\boldsymbol{X_L}, \boldsymbol{Y_L})_{\boldsymbol{ini}} \cup (\boldsymbol{X_s}, \boldsymbol{Y_o})$

The first step of the proposed scheme is to use the initially selected and annotated data in order to train the network. Note that we have used the deep parallel residual networks presented in section 3.2.2 in order to boost the discrimination potential of this initial supervised training procedure.

After training the networks, we use the initial group of data in order to generate their relevance maps. We can use the DeepTaylor or LRP maps. In our experiments, we mostly used the DeepTaylor maps. Those generated maps will be considered as our references (data). After having generated the relevance maps of the data, we construct the clusters of those relevance maps. In this step, we apply agglomerative clustering using the relevance maps as the input data. We stopped the linkage, of course, when we have the number of clusters corresponding to the number of classes that we have in our dataset.

After we have the clusters, we can compute their centers. Instead of just using the mean, we used the medoid, which means that we have found the points, in the clusters, for which the sum of distances with all the other members of the clusters was the minimum. These clusters' centers will be considered as the representations of the constructed clusters. In other words, our next evaluation of the uncertainty according to a given cluster will be made with having the cluster's center representing the instance of the cluster.

The next step is obviously the fact of using the whole set of the unlabeled data as the input of the previously trained network. With the trained network in hands, we can generate the relevance maps of the unlabeled set. The most important part of the process is to compute the confidence by utilizing these generated relevance maps. In our proposed method, the confidence is redefined by utilizing the distance between a given map and the clusters' centers computed previously. This distance is used as the equivalent of the network's outputs in the conventional active learning selection methods. Thus, we can say that, in our method, the confidence is a function of the distance between the relevance maps and their clusters' centers.

Following this redefinition, the different uncertainty measures in Equation (1), Equation (2) and Equation (3), which were defined in section 2.4, are rewritten as follows. For a given instance X_i, the entropy is given by the following expression:

$$entropy(X_i) = \sum_{j=1}^{k} d_k \log d_k, \tag{16}$$

where $d_k = d[\boldsymbol{\mathcal{F}_{rel}}(X_i), C_k]$. As explained in Table 3, $\boldsymbol{\mathcal{F}_{rel}}(X_i)$ represents the relevance map of the data X_i. C_k represent the clusters' centers computed in step (4), with k being the number of clusters that we have. The expression $d[\boldsymbol{\mathcal{F}_{rel}}(X_i), C_k]$ denotes the k different distance values that separate the relevance map and the k clusters' centers. Hence, in Equation (16), the distances d_k replace the probabilities used in Equation (1).

Following the similar idea, the confidence C from Equation (2) can be rewritten as

$$C(X_i) = \underset{k}{\operatorname{argmax}}\, d_k. \tag{17}$$

Also, using the margin-sampling idea, the confidence in Equation (3) is redefined as

$$C(X_i) = max_1(d_k) - max_2(d_k), \tag{18}$$

where $max_1(d_k)$ and $max_2(d_k)$ are the highest and second distances, respectively.

As we can clearly remark in Equations (16) - (18), the probabilities generated by the model are completely replaced by the distances between the new map and the clusters' centers. A data is then considered to be uncertain for the model if its relevance map happens to be at quasi equidistance to all the clusters' centers.

The next step will consist of ranking the data according to their confidence. The data with the lowest confidence are selected and given to the oracle in order to be annotated. The explainability of this selection process is evident in the sense that the data selected will be sufficiently different from the others. A relevance map that is very different with the other maps reveals a data that is sufficiently different, which is, in fine, a data for which a model will be uncertain. Note that this explainability is not manifest in the conventional selection process.

The next step is to mix the newly annotated data with the initial labeled set constructed at the beginning of the process. After this step, we will obtain a newly labeled set, which will be bigger than the initial set. The next step in the process is to use this newly labeled set in order to re-train the model. As for the conventional method, this iterative selection process can continue until we label all the available data. In our case, we stop the labeling process when we reach the sampling budget. In Fig. 23 and Table 3, all the different steps explained here are numerated from step (1) to step (11).

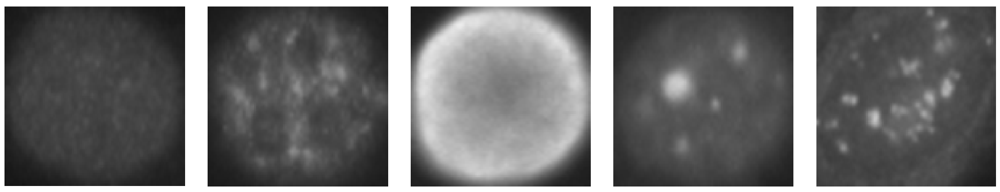
IV. EXPERIMENTAL RESULTS AND DISCUSSION

In this chapter, we discuss about the datasets used for the experiments and present the obtained results. Results are divided into two sections. The first presents the datasets and results of the deep clustering method. The section presents also the datasets and the results for the active learning approach.

4.1 Deep clustering results

4.1.1 Datasets and experimental setups

We first show the results obtained using the SNPHEp-2 dataset. The description of this dataset can be found in [69]. The SNPHEp-2 dataset comprises five types of cells: the homogenous cells, the coarse speckled cells, the fine speckled cells, the nucleolar cells and the centromere cells. This dataset contains two levels of fluorescence intensity: positive and negative intensities. Some examples from this dataset are depicted in Fig. 24 and we can remark that the different fluorescence intensities increase the intra-class variations of the dataset. In Fig. 24 (a), we have the positive illumination images and Fig. 24 (b) we have the negative (or intermediate) illumination images. We can remark how the differences between the images belonging to the same cellular type but having different types of fluorescence illumination are quite manifest, which demonstrate the intra-class variations.


(**a**)

(**b**)

Fig. 24. Example images from the SNPHEp-2 dataset. (**a**) The positive fluorescence illumination images. (**b**) The negative fluorescence illumination images. In (**a**) and (**b**), from the left to the right: the Homogeneous, the Coarse Speckled, the Fine Speckled, the Nucleolar and the Centromere.

The dataset contains 1,884 HEp-2 cell images. The images were all extracted from 40 different cell specimens. From the 40 specimens, 20 were used for the training sets and the remaining 20 were used for the testing sets. In total, there are 905 and 979 cell images for the training and testing sets, respectively. Each set (training and testing) contains five-fold validation splits of randomly selected images. In each set, the different splits are used for cross-validating the different models, each split containing 450 images approximatively. The dataset can be downloaded at http://staff.itee.uq.edu.au/lovell/snphep2/. During the experiments, the images were up-scaled to 112x112 in order to use them in our network. Note that the original images' size varies between 60x60 and 90x90.

In order to improve the learning capability of the network, data augmentation was applied over the dataset. In every splitting set of the training data, the cells were rotated for 360° with the step of 18°, as done in [80,89]. Which means that the original training was expanded by a factor of 20, a 360° quadrant containing 20 portions of 18°. We found that augmenting the training set really improves the accuracy over the testing set. All the results shown subsequently are those obtained after applying data augmentation in this form.

As explained in the previous section, the global loss, defined in Equation (8), is minimized by updating the network's parameters. In addition to the weights and biases usually associated with the DNN, we have the clusters' centroids, which need also an update at every epoch during the training. As usually done with the k-means process, we need some initial centroids in order to launch the updating process. Instead of a random setting, we use a pre-training of the DCAE in order to generate the first clusters, as done in [15-17,102]. The idea is to firstly use the DCAE for generating a preliminary distribution of the data, and then, utilize the computed features for finding the initial centroids. This pre-training can be seen as setting γ to be zero in Equation (8), which means that the learning is done by using only the reconstruction loss L_r. After generating the initial centroids, we perform the learning as proposed here by using the global loss with $\gamma > 0$. This second training process using the global loss is subsequently referred as the "global learning".

As explained in the previous section, the parameter γ is very important because it balances the importance of the clustering loss L_c in the global loss. As also previously mentioned, the reconstruction loss itself can help to preserve the local structure of the data. That is the reason why, as explained in [15], the coefficient γ is better to be less than 1 in order to permit the L_r to have more importance than L_c in the global loss. In our experiments, by using cross-validation, the best results were obtained with the value of γ being 0.1. Note that this value of γ provides also the best results in [15] while dealing with images.

The parameters are updated by using backpropagation. The gradients of the global loss can be derived as

$$\frac{\partial L}{\partial (Z,c)} = \frac{\partial L_r}{\partial (Z,c)} + \gamma \frac{\partial L_c}{\partial (Z,c)}, \tag{19}$$

where Z represents the set of weights and biases, and c represents the clusters' centroids. Except for γ, for which different values (0 and 0.1) were used for the pre-training and the global learning, all the other hyperparameters were used similarly for the two steps. The learning rate was set to be 0.01 and the size of the mini-batch was 128. The momentum was set to be 0.9 and the weights initialization follows the process presented by He et al. [111]. The pre-training was done with 200 epochs, while the global learning was trained for 450 epochs. The experiments were done with the use of MATLAB R2019b and performed on a computer with a Core i7 3.40 GHz processor and 8 GB of RAM. A GPU implementation was used with a NVIDIA GeForce GTX 1080 Ti with 11,264 MB of memory.

Specific metrics are usually adopted for evaluating the clustering performance. These metrics include the normalized mutual information [112], the adjusted Rand index [113] and the clustering accuracy (ACC) [112]. Because we aim to compare our method with the state-of-the-art HEp-2 cell classification methods that all utilize supervised learning, we only use the ACC for showing the results, since it is equivalent to the accuracy as it's measured in supervised learning. Since we have the actual labels of the data, we can evaluate the method in the same way as it is done with the classification (supervised). A data is considered to be well classified if the cluster to which it was assigned by the network corresponds to its actual label. Consequently, confusion matrices are used to show the results. The sensitivity (true positive rate) and the specificity (false positive rare) of every single cellular type can be derived from the confusion matrices.

The results are shown for three different cases:

- The first case, referred as "**case-1**", consists of using the network without the proposed techniques for assuring a better reconstruction, which means that we rely only on the reconstruction loss in order to assure a better local preservation, as proposed in [15,102].
- The second case, referred as "**case-2**", consists of using the pooling indices and the copy-and-concatenation techniques, as shown in the network depicted in Fig. 12.
- And the final case, referred as "**case-3**", consists of using the proposed attention-based network, which also incorporates the pooling indices and the copy-and-concatenation techniques at the same time, as illustrated in the network shown in Fig. 15.

4.1.2 Results for Case-1

For the **case-1**, we use the network as defined in Table 1, without applying the techniques proposed here for assuring a good local preservation of the pixels, thus, a better reconstruction. After the pre-training ($\gamma = 0$), we use the pre-trained DCAE in order to perform the global learning ($\gamma > 0$). Note that the features learned by the DCAE, the same goes for the clusters' centroids, are 512-dimensional vectors, as we can see in Table 2. For visualizing these high-dimensional vectors, we have applied principal component analysis (PCA) [6]. For all the features' visualization shown here, PC_1 and PC_2 are, respectively, the first and second axis of the PCA-space. The projections of the clusters learned by the **case-1**'s DCAE are shown in Fig. 25.

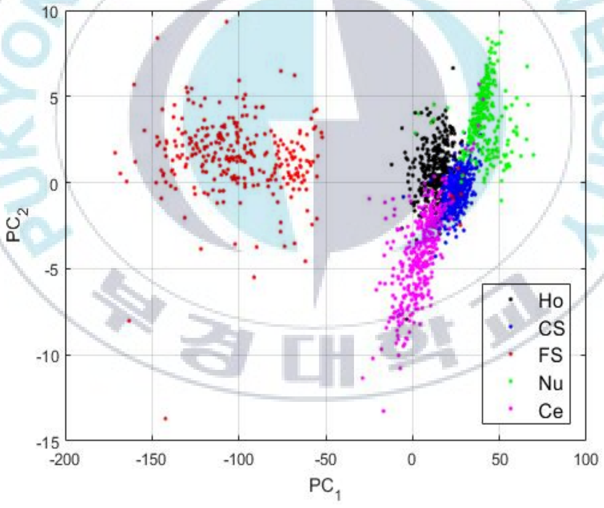
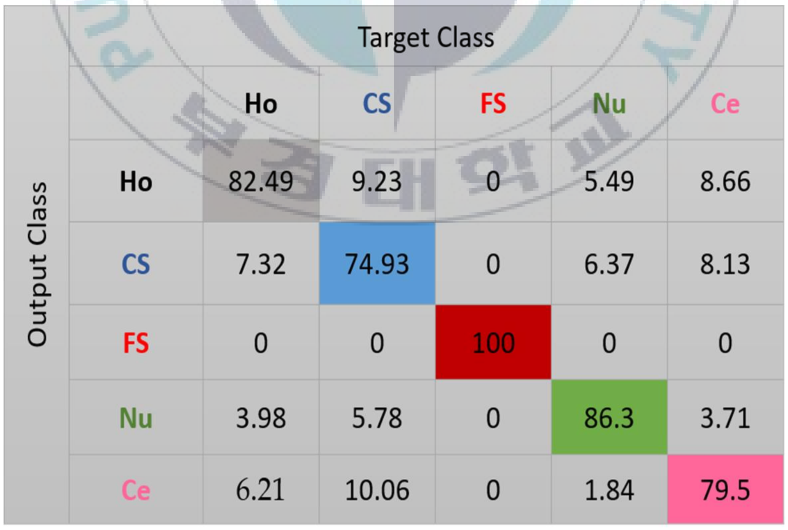


Fig. 25. Visualization of the features learned by the DCAE of case-1. "Ho", "CS", "FS", "Nu", and "Ce" represent the homogeneous, the coarse speckled, the fine speckled, the nucleolar and the centromere, respectively. The percentages of the variance explained are respectively of 99.23 and 0.42 for the first and second principal components.

The first observation from the projections shown in Fig. 25 is that the network has clearly learned to distinguish the different cellular types. Two main clusters are visible in Fig. 25: the cluster formed by the fine speckled cells and the one formed by the remaining cells. The fine speckled cells, as we can see in Fig. 24, are remarkably distinguishable from the others. The fine speckled cluster itself contains relatively two subgroups, which represent the two different fluorescence illumination (intensity levels). If we can expect from these features that the majority of the fine speckled will be well classified (assigned to the cluster that corresponds to their true labels), the situation is more complicated for the other clusters that share many similarities.

The results for the **case-1**'s network are shown in the confusion matrix depicted in Fig. 26. As expected, all the fine speckled cells were assigned to the right cluster, the one that corresponds to their true label. The remaining cells all suffer from a lack of clear distinguishability between them, which really decreases the accuracy of the cluster's assignment. Note that, as for the projections shown in Fig. 25, every cluster is represented by its corresponding color in Fig. 26: the black for the homogeneous, the blue for the coarse speckled, the red for the fine speckled, the green for the nucleolar and the magenta for the centromere. Major confusions in Fig. 26 concern the coarse speckled cells and the centromere, whose assignment accuracy is 74.93 % and 79.5 %, respectively. The total accuracy of the **case-1**'s DCAE is about 84.64 %.



		Target Class				
		Ho	CS	FS	Nu	Ce
Output Class	Ho	82.49	9.23	0	5.49	8.66
	CS	7.32	74.93	0	6.37	8.13
	FS	0	0	100	0	0
	Nu	3.98	5.78	0	86.3	3.71
	Ce	6.21	10.06	0	1.84	79.5

Fig. 26. Confusion matrix for case-1. "Ho", "CS", "FS", "Nu", and "Ce" represent the homogeneous, the coarse speckled, the fine speckled, the nucleolar and the centromere, respectively.

4.1.3 Results for Case-2

For the **case-2**, we use the network as depicted in Fig. 15. The features are shown in Fig. 27. By comparing the projections shown in Fig. 25 and the ones shown in Fig. 27, we can notice how the four other clusters have become distinguishable. The fine speckled cells are still clustered to their own sub-space, as for the projections in **case-1**. This shows that when the reconstruction process from the DCAE tends to preserve the local structure of the original images in the best way possible, the features learned by the DCAE are also better. The results for the cluster's assignment of **case-2** are shown in the confusion matrix in Fig. 28.

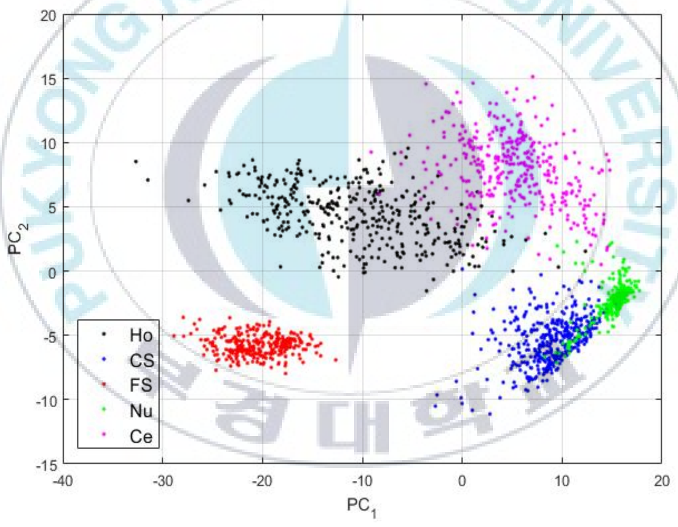
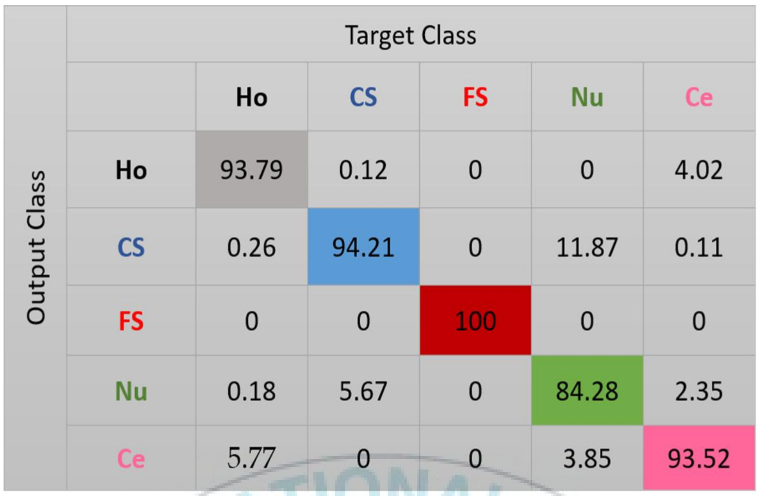


Fig. 27. Visualization of the features learned by the DCAE of case-2. "Ho", "CS", "FS", "Nu", and "Ce" represent the homogeneous, the coarse speckled, the fine speckled, the nucleolar and the centromere, respectively. The percentages of the variance explained are respectively of 83.06 and 16.38 for the first and second principal components.



Output Class \ Target Class	Ho	CS	FS	Nu	Ce
Ho	93.79	0.12	0	0	4.02
CS	0.26	94.21	0	11.87	0.11
FS	0	0	100	0	0
Nu	0.18	5.67	0	84.28	2.35
Ce	5.77	0	0	3.85	93.52

Fig. 28. Confusion matrix for case-2. "Ho", "CS", "FS", "Nu", and "Ce" represent the homogeneous, the coarse speckled, the fine speckled, the nucleolar and the centromere, respectively.

In Fig. 28, we see how most of the confusions between the cells have completely disappeared. The remaining wrong assignments concern mostly the nucleolar cells. As we can see in the projections shown in Fig. 27, the nucleolar patterns (shown in green) tend to share the same clustering subspaces with the coarse speckled and the centromere. This is explained by the similarities shared by these three cellular types in terms of the shape and intensity, as we can notice in Fig. 24. Specifically, 11.87% of the nucleolar cells are clustered as coarse speckled. In Fig. 27, the features from the nucleolar and coarse speckled (green and blue) are largely mixed. And also, 5.67% of the coarse speckled are misclassified as nucleolar. Besides that, the clustering accuracy increases from 84.64% (for case-1) to 93.16%. Which clearly indicates that the more we can encourage the network to preserve the local structure of the original images, the more the features learned by the network will contain the distinctive characteristics of the cellular images.

4.1.4 Results for Case-3

The **case-3** consists of using the attention-based network as designed in Fig. 15, where the pooling indices storage technique is mixed with the copy-and-concatenation process and, furtherly, the three different streams all learn how to boost the reconstruction accuracy. The copy-and-concatenation operations allows the network's decoder to retrieve the entire spatial information lost during the down-sampling operated in the encoder. When mixed with the pooling indices storage, not only the local structures (close neighborhood) of the original pixels are well preserved, but also, the global spatial information, concerning the whole image, is retrieved during the reconstruction. Although the two added streams tend to add complexity for the computations, they allow to smooth and improve the reconstruction quality.

The features for this case are shown in Fig. 29. We can notice how, compared with the projections in Fig. 27, the clusters continue to be more precise. Some of the nucleolar cells continue to be mixed with the centromere but, globally, the confusions between the cells are really diminished. The cluster's assignment results are shown in details in the confusion matrix depicted in Fig. 30. As we can remark, most of the misclassifications have disappeared. The centromere cells are still mixed with some homogeneous (2.23%) and nucleolar (4.08%). The most outstanding improvement comes from the nucleolar cells. The totality of the confusion between the nucleolar and the coarse speckled is vanished. Only remained some misclassification between the nucleolar and the centromere. The total accuracy of the clustering assignment is 97.59% for the results shown in Fig. 30.

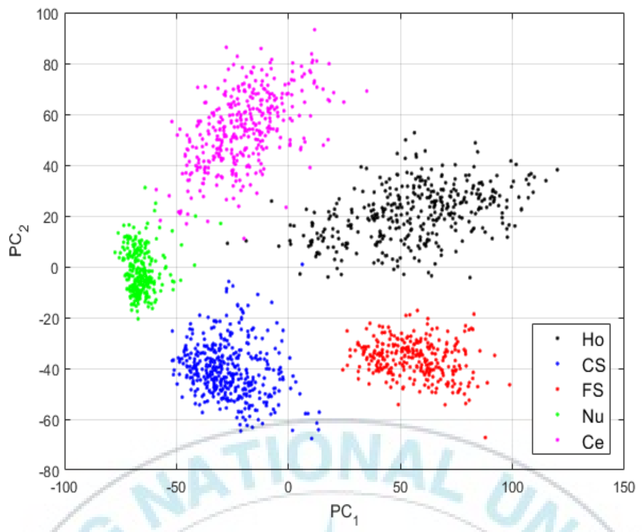
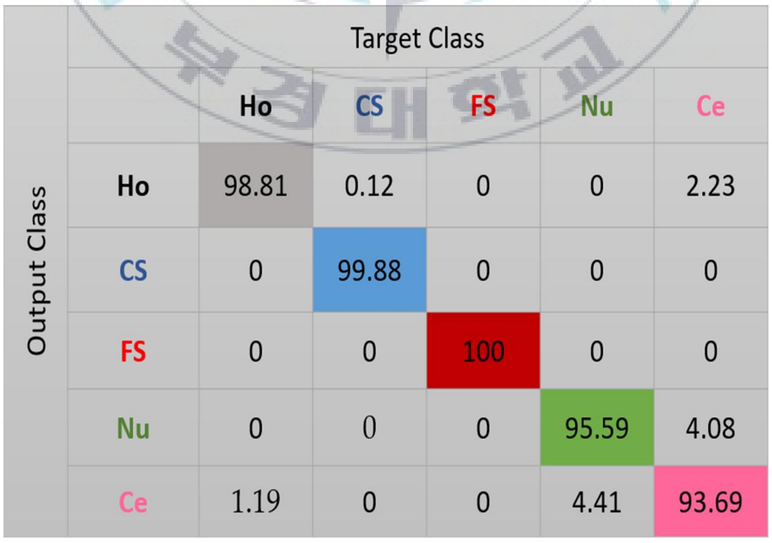


Fig. 29. Visualization of the features learned by the DCAE of case-3. “Ho”, “CS”, “FS”, “Nu”, and “Ce” represent the homogeneous, the coarse speckled, the fine speckled, the nucleolar and the centromere, respectively. The percentages of the variance explained are respectively of 56.86 and 33.61 for the first and second principal components.



		Target Class				
		Ho	CS	FS	Nu	Ce
Output Class	Ho	98.81	0.12	0	0	2.23
	CS	0	99.88	0	0	0
	FS	0	0	100	0	0
	Nu	0	0	0	95.59	4.08
	Ce	1.19	0	0	4.41	93.69

Fig. 30. Confusion matrix for case-3. “Ho”, “CS”, “FS”, “Nu”, and “Ce” represent the homogeneous, the coarse speckled, the fine speckled, the nucleolar and the centromere, respectively.

Fig. 31 shows the summary of the results obtained by the three different networks in terms of the accuracy. The results in Fig. 31 also demonstrate the effectiveness of using data augmentation when the training data is really small. As we can remark, the results remain poor with the three networks when we use the original data without any augmentation during the training. On the other hand, we can see how the accuracy is improved when data augmentation is applied. We show the results for $\theta = 36$, which means that the rotation is done with a step of 36 degrees, increasing the number of training data by a factor of 10, and for $\theta = 18$, which increases the training data in a factor of 20. All the three network provide the best results with $\theta = 18$, as we can see in Fig. 31.

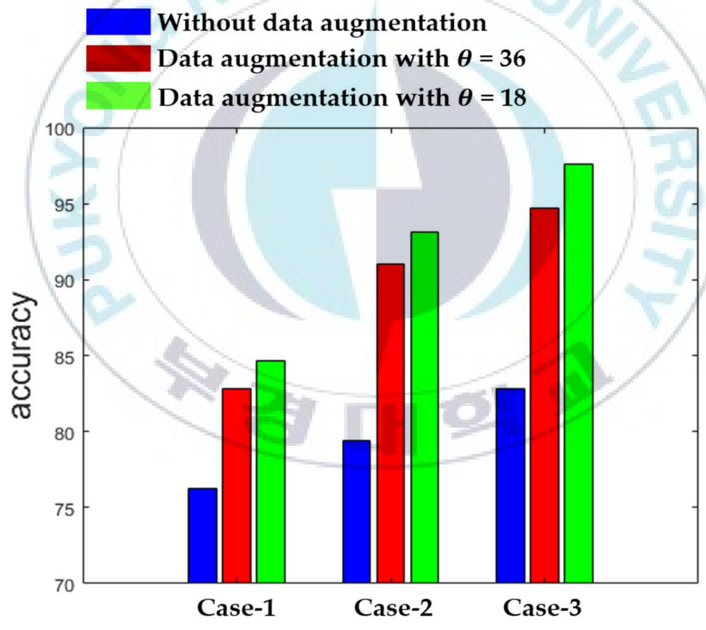
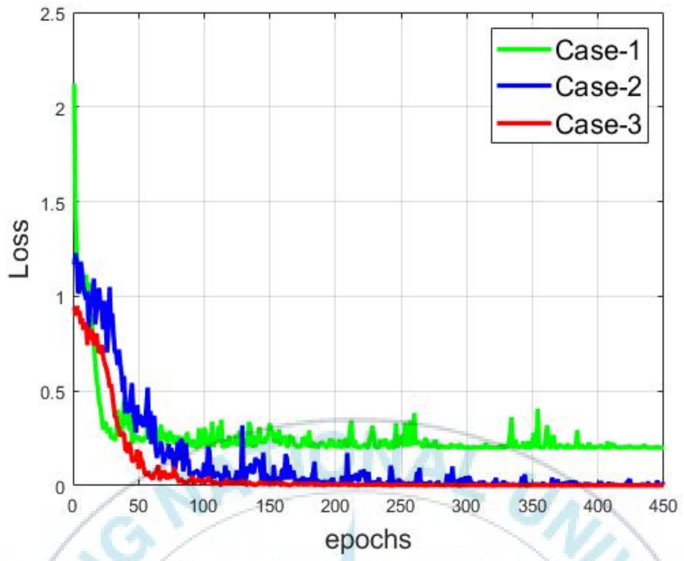
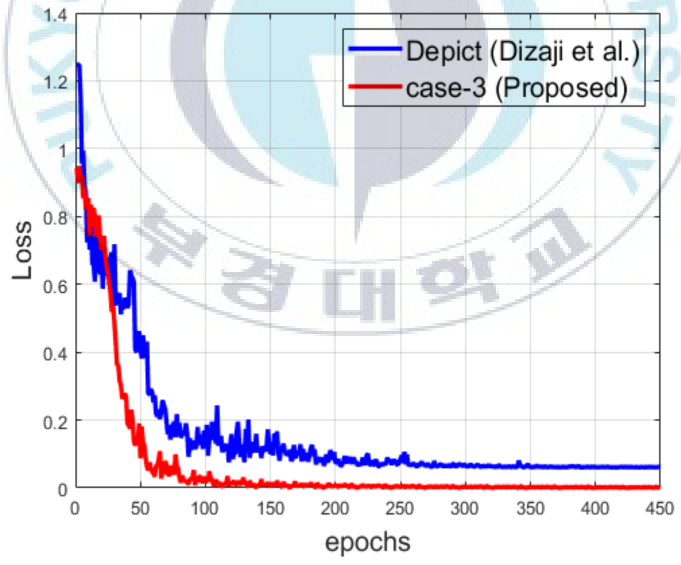


Fig. 31. Clustering accuracy of the three networks.



(a)



(b)

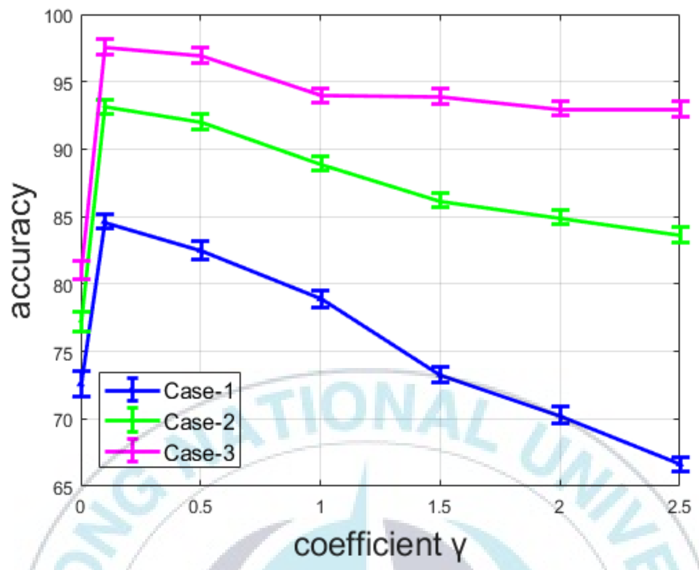
Fig. 32. The global loss of the networks. (a) Comparison between the three proposed networks. (b) Comparison between the proposed **case-3** and the dual autoencoder in [21].

Another comparison between the three networks is provided in Fig. 32 (a), where we show the evolution of the global loss, which encapsulates the reconstruction and the clustering losses. Among the three cases, the network from **case-3** provides the most minimal loss, since the copy-and-concatenation mixed with the three streams system process alleviates the reconstruction process by allowing the network to preserve at the most the local structure of the data. We can remark that the loss is more smoothed compared to the two other cases. The local structure preservation permits a faster reconstruction compared to the others. Note that the loss is also consequently diminished in the **case-2**. With the results shown in Fig. 32 (a), we can notice the improvement that occurs when the reconstruction is made in the best way possible (**case-2** and **case-3**).

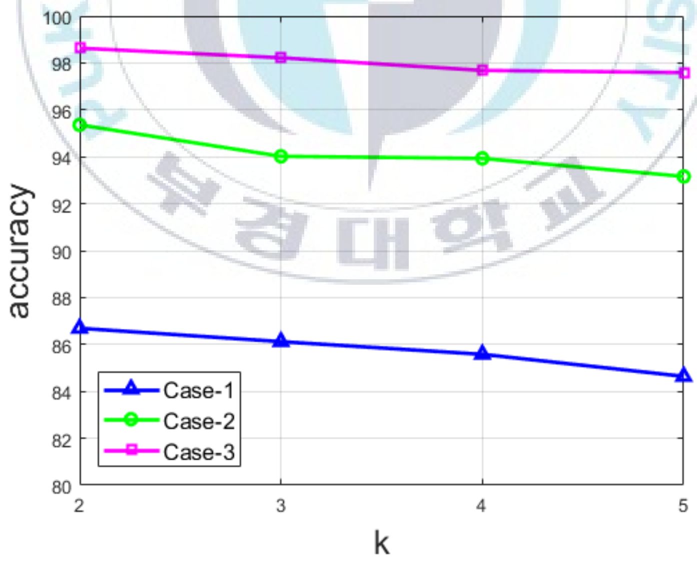
The results in Fig. 32 (b) is a comparison between the reconstruction of our **case-3** network and the network proposed by Dizaji et al. [21] while using the cellular images. Note that our proposed attention-based network outperforms the dual autoencoder in terms of the reconstruction.

Another finding concerns the parameter γ that controls the balance between the reconstruction and the clustering losses. As mentioned before, the best results were obtained with the value of 0.1. We found that the accuracy decreases every time the coefficient γ increases. Which means that, when the clustering loss tends to overshadow the reconstruction loss, the features lose their distinctive characteristics, which encourages misclassification during the assignment. This fact also, besides the results demonstrated above, contributes to the corroboration of the assumption that the preservation of the local structure of the images helps to produce better features.

Fig. 33 (a) shows the variations of the clustering accuracy with different values of the coefficient γ for the 3 different networks. Note that, here, the situation where $\gamma = 0$, as explained above, corresponds to the pre-training of the DCAE, where only the reconstruction loss is used in order to generate the initial features that will be used to compute the initial centroids. We can notice that, for all the networks, the accuracy for the pre-training ($\gamma = 0$) is very low. With $\gamma > 0$, all the networks provide their best results with the value of 0.1 and their accuracy starts to decrease with every $\gamma > 0.1$.



(**a**)



(**b**)

Fig. 33. Accuracy of the three networks: (**a**) with different values of coefficient γ, and (**b**) with different values of k (number of clusters).

The most important point is that the accuracy decreases differently for the three networks. It decreases rapidly with the network from **case-1**, a bit slower with the one from **case-2** and it seems to be stable with the network from **case-3**. For the **case-1**, there is an important decline of the accuracy when the coefficient $\gamma \geq 0$. This network does not contain any element in its design that can lessen the loss of the local structure, besides, of course, the reconstruction loss. It clearly appears that the clusters lose their efficiency when the clustering loss completely overshadows the reconstruction loss. On the other hand, the **case-2** and **case-3** networks are purposely designed in order to minimize the loss of the spatial details and preserve the local structure of the original images. Both suffer a decrease of the accuracy with $\gamma \geq 0$, but, the reduction is minimized.

Again from Fig. 33 (a), we can remark that the accuracy of the network is not really dependent to the value of the coefficient γ for the **case-3**'s network. Even though there is an evident decrease, the variation is not very noticeable, as for the two other cases. This comes from the fact that the two combined techniques used for the **case-3** allow the network to continue of assuring a better reconstruction even when the clustering loss tends to overshadow the reconstruction loss. Which means that the **case-3**'s network can still assure a better reconstruction even without giving the reconstruction loss a solid importance in the global loss. With these results, we can affirm that the quality of the reconstruction process affects the quality of the produced features, and thus, the final clustering's accuracy.

In Fig. 33 (b), we show the variation of the accuracy by changing the number of clusters. Note that the variation is really small, since the value k is also small. Therefore, as we can see in Fig. 33 (b), there is no noticeable change when the number of clusters changes.

In Table 4, we show the results of the different methods in the literature. We separate the methods in 3 groups: the supervised learning handcrafted features, the supervised deep learning methods and the unsupervised deep learning method. The supervised learning based on the handcrafted features have achieved, respectively, 80.90%, 82.50%, and 85.71%. The method in [80], which is one of the first ever to apply deep learning for the HEp-2 cell images classification, reaches 86.20%. Note that the datasets available at that period were not diversified enough to perform well with deep learning.

The deep learning-based method in [81] has a quasi-identical structure with the one used in [80] but, unlike in [80], they have applied many different techniques for data augmentation, allowing them to perform at 88.37% of accuracy. In Table 4, in the last three lines of the "supervised deep learning", we can see the performance of the actual state-of-the-art methods. Of course, as discussed before, all of them utilize the supervised learning approach.

Table 4. Comparative study for the SNPHEp-2 dataset.

Method	**Description**	**Accuracy**
Supervised learning Hand-crafted features	Texture features + SVM [108]	80.90%
	DCT features + SIFT + SVM [69]	82.50%
	LBP + SVM [71]	85.71%
Supervised Deep Learning	Simple CNN [80]	86.20%
	Simple CNN [81]	88.37%
	CNN with Deep Residual Inception Module [86]	95.61%
	CNN using Cross-modal transfer learning [100]	95.99%
	CNN with a Deep-Cross Residual Module [89]	96.26%
Unsupervised Deep Learning	DCAE with an embedded clustering layer (**case-1**)	84.64%
	DCAE with an embedded clustering layer (**case-2**)	93.16%
	Pseudo-classification using CNN [20]	94.19%
	Dual autoencoder with both noisy and clean encoders [21]	95.78%
	DCAE with an embedded clustering layer (**case-3**)	97.56%

We can see in Table 4 that the **case-1**'s network performs at the same level with the hand-crafted features. When we apply the proposed approach (**case-2** and **case-3**), the proposed method performs at the same level with the state-of-the-art supervised deep learning methods (**case-2**) and even slightly better (**case-3**). The two other deep clustering methods reach 94.19% for the pseudo-classification using the CNN and k-means in order to generate the "pseudo-labels" proposed by Caron et al. [20], and 95.78% for the dual autoencoder proposed by Dizaji et al. [21].

Table 5. Comparative study for the 13A dataset.

Method	**Description**	**Accuracy**
Supervised Learning Hand-crafted features	Texture features + SVM [108]	71.63%
	DCT features + SIFT + SVM [69]	74.91%
	LBP + SVM [71]	79.44%
Supervised Deep Learning	Simple CNN [80]	97.24%
	Simple CNN [81]	98.26%
	CNN Deep Residual Inception Module [86]	98.37%
	CNN using Cross-modal transfer learning [100]	98.42%
	CNN with a Deep-Cross Residual Module [89]	98.82%
Unsupervised Deep Learning	DCAE with an embedded clustering layer (**case-1**)	81.23%
	DCAE with an embedded clustering layer (**case-2**)	86.42%
	Pseudo-classification using CNN [20]	91.15%
	Dual autoencoder with both noisy and clean encoders [21]	92.69%
	DCAE with an embedded clustering layer (**case-3**)	94.89%

The 13A dataset was also used in order to test the proposed method. This dataset has a far bigger number of data (13,596) compared to the first one and seems to be much easier to handle by the deep learning-based methods. Cross-validation was performed using the protocol used in [80]. 80 % of the data were used for training and validation (using a 64%-16% split) and the remaining 20% were utilized for testing. We have applied data augmentation in the same way as described previously.

Compared to the first dataset, the handcrafted features perform very poorly on this one. The reason is that the big amount of images from this dataset provides a very strong learning capability to the deep learning methods while it brings more complexity to the handcrafted ones. For this dataset, we show the results just as they were reported by the authors in their works. We have experimented only the handcrafted features' works because all of them were proposed at the time when this dataset was not available. The results are shown in Table 5.

Note that, as we can notice in Table 5, all the state-of-the-art supervised deep learning methods perform similarly on this dataset. Our method (**case-3**) also performs at the same level with the supervised learning methods in terms of the accuracy, but, with the advantage of being entirely unsupervised. The two other deep clustering methods also perform well, reaching a quasi-same level with the supervised learning methods. It should also be noted that this dataset contains far more data compared to the first one, which slightly adds more complexity for the unsupervised learning methods.

Because most of the deep clustering methods in the literature are not proposed for the cellular images, we have conducted a comparative study using the popular handwritten digits (MNSIT) dataset between our threes cases, the method using the CNN in a pseudo-classification, and the dual autoencoder. Using the MNIST, the three cases perform at 93.46%, 95.63% and 97.45%, respectively. The pseudo-classification method performs at 91.38%, while the dual autoencoder reaches 96.50%. Note that all the deep clustering methods perform at a quite pleasant level for the handwritten digits' images.

4.2 Active learning results

For the active learning part, results are presented in this way:

- We first show the results using the proposed dynamic learning system. Since this method was specifically designed for the HEp-2 cell images, we show the results only for these images in the first part. These results are obtained by using the conventional active learning-based selection but, with the proposed parallel deep networks. The goal of these experiments is to demonstrate the effectiveness of (1) the proposed deep parallel residual networks, (2) the cross-modal transfer learning and (3) active learning in general for the HEp-2 cell classification.

- In the second part (section 4.2.6), we present the results using our proposed explainable active learning. Since this method utilizes XAI, we start by showing how we selected the relevance map method. After this step, we show the results obtained by using our proposed selection method on the cellular images. Then a comparative study is conducted using both the cellular images and the CIFAR-10 dataset.

4.2.1 Datasets and initial setups

The deep residual networks were first trained using a relatively small dataset, as mentioned before. We have adopted the SNPHEp-2 dataset here for this initial learning process. This dataset was presented in details at the beginning of the previous section and we have shown some example images from it in Fig. 24.

Instead of using the ICPR2012 for the initial learning as done in [100], the choice of using the SNPHEp-2 dataset was justified by the similarities between this dataset and our targeted dataset. The targeted dataset is the large-scale HEp-2 (LSHEp-2) dataset, introduced by Qi et al. [114]. This dataset contains far more images (63,445) than the 13A dataset (13,596 images) presented in the previous section. The reason why we adopted the LSHEp-2 dataset is that we aim to test the effectiveness of the proposed active learning scheme on a really big dataset for which labeling can really represent a burdensome task.

Furthermore, this dataset is more complex in terms of intra-class variations and heterogeneity compared to the others. The description of the LSHEp-2 dataset can be found in details in [100] and it can be downloaded at http://qixianbiao.github.io/HEp2Cell/. Similar to the 13A dataset, it contains six classes: Homogeneous, Speckled, Nucleolar, Centromere, Nuclear membrane, and Golgi. Some examples of this dataset are shown in Fig. 34.

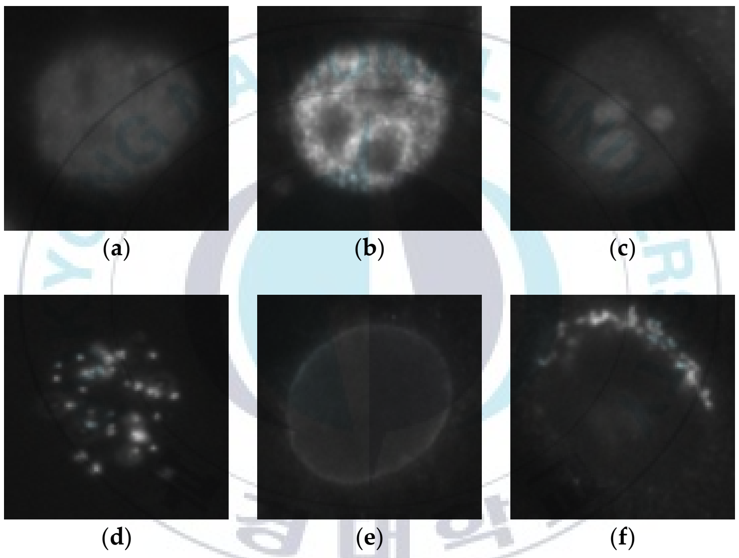
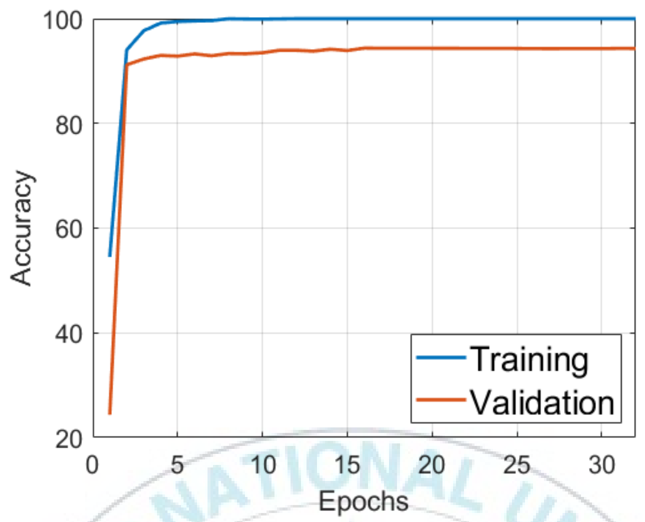


Fig. 34. Example images from the large-scale HEp-2 dataset. (**a**) the Homogeneous, (**b**) the Speckled, (**c**) the Nucleolar, (**d**) the Centromere, (**e**) the Nuclear membrane, and (**f**) the Golgi.

By comparing the images in Fig. 24 and Fig. 34, we can remark that the two datasets share many similarities. We can expect that our networks will learn the general features shared by these two sets of images, which will allow us to only update the task-specific layers located at the end of the networks. In fact, two big changes can be remarked between the SNPHEp-2 and the LSHEp-2 datasets: firstly, the two speckled (fine and coarse) cells from the first were mixed to form only one cell type, the speckled cells, in the second. Secondly, the Golgi are absent from the SNPHEp-2. During the transfer learning process, we will remove the last layer containing 5 neurons and replace it by another containing 6 neurons. In this part, all the experiments were performed using TensorFlow on a computer with a Core i7 3.40 GHz processor, 8 GB of RAM, and a NVIDIA GeForce GTX 1080 Ti GPU.

For the initial learning process, the hyperparameters were selected via cross-validation using the 5 different validation folds of the SNPHEp-2 dataset. The original images have different sizes (average around 90×90) and were all up-sized using bicubic interpolation to 112×112 in order to fit into our designed architecture. Note that after the DWT decomposition, the coefficients at the first level have all the size of 56×56. In order to maximize the learning capacity of the networks, data augmentation was applied over the SNPHEp-2 dataset. It consists of cell rotation, with a step of 18° in a quadrant of 360°, as proposed in [80,89]. This rotation increases the original training set by a factor of 20.

The learning rate is set to be 0.001 and training is terminated when the validation loss does not surpass the reached minimum 5 times in a row. For the initial learning (with data augmentation), 32 epochs were necessary to terminate the training process (see Fig. 35 (a)). The classification results of this initial learning are shown in the confusion matrix depicted in Fig. 35 (b). The accuracy over the validation set, as we can see in Fig. 35, was about 94 %. We recall here that the purpose of this initial learning is just to generate a pre-trained model that will be used later for the transfer learning. For further details about the effectiveness of the dynamic learning afforded by the wavelet coefficients, readers are invited to check our previous work [109] where we have presented a detailed discussion about it.



(**a**)

Output Class \ Target Class	Homo	Coarse	Fine	Nucl	Centro
Homo	**94.96**	1.63	2.86	0.53	0
Coarse	1.45	**91.23**	1.02	2.60	3.49
Fine	2.82	1.55	**95.85**	0.10	0
Nucl	0.41	2.18	0.24	**94.60**	1.59
Centro	0.36	3.41	0.03	2.17	**94.92**

(**b**)

Fig. 35. Results of the initial learning process: (**a**) the progression of the accuracy of training and validation; (**b**) the confusion matrix (total accuracy of 94.31%). “Homo”, “Coarse”, “Fine”, Nucl”, and “Centro” stand for Homogeneous, Coarse Speckled, Fine Speckled, Nucleolar and Centromere, respectively.

After we have our pre-trained model in hands, we can utilize it for the active learning over the LSHEp-2 dataset. For all the fine-tuning procedures, we have fixed all the layers before the second residual block, which means from the first convolutional layer to the second pooling layer. The second residual block is set to be trainable because we want the networks to extract features that are specific to our main dataset before the feature fusion (layer concatenation). The final layer was changed to have six neurons, according the six classes of the LSHEp-2 dataset. The same learning approach was used for all the fine-tuning processes: a learning rate of 0.001 was utilized, training is stopped when the loss does not decrease five times in a row.

As said before, the LSHEp-2 dataset contains 63,445 images. A 80%-20% splitting is performed, which gives 50,758 images for training, and 12,687 for testing. The labeling limitations that we impose to ourselves only concern the training set (the testing set is just used for validation, not for fine-tuning). We principally tested our method with the limitation of being able to annotate only 20% of the training set, which gives a total of 10,152 images. The value *k* is set to be 1,500, which means that 1,500 images (around 15% of the 10,152) are first selected randomly in order to perform the first fine-tuning. And then, the value *m* is set to be 1000, which means that we select the first 1000 data in the ranking performed in step 4 (see Table 2) in every labeling iteration. We stop the labeling process after we annotate the totality of the 10,152 images. Note that the same kind of process is repeated for any level of limitation (the number *k* being 15% of the limitation, and *m* being 10%).

The results are shown in two categories: the results without cross-modal transfer learning and the ones with cross-modal transfer learning. The first category designates the case where we do not use the initial learning for building the pre-trained model. We just train the networks by using directly the first *k* images from the main dataset. The second category designates the proposed scheme, where initial learning with a small dataset is used before fine-tuning with the main dataset. In every category, we show two cases for the results: results using random sampling and results using active learning-based sampling. In other words, and with the case of 20% of limitation, random sampling designates the fact of selecting randomly 20% of the training images in order to train the networks while active learning-based sampling designates the fact of using active learning techniques for the selection of the 20% of images.

Note that for all the cases where active learning is involved, we do not show the loss and accuracy progression since several different learning procedures are conducted in every labeling iteration (many fine-tunings). In this case, showing the loss and accuracy evolution is meaningless. On the other hand, these evolutions are shown for the cases that do not involve active learning, where only one single training procedure is performed. For simplicity, the different cases are designated by their short names shown in Table 6.

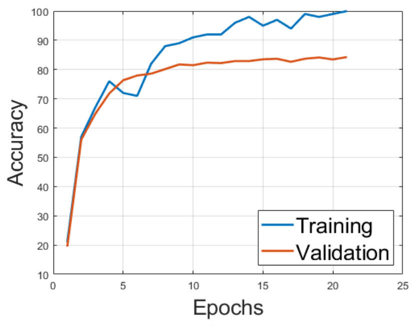
Table 6. The different cases used during the experiments.

Case name	**Comments**
RS (random sampling)	No initial learning, and selection using *random sampling*.
AL (active learning)	No initial learning, and selection using *active learning*.
IN-RS (random sampling with cross-modal transfer learning)	Initial learning involved, and selection using *random sampling*.
IN-AL (active learning with cross-modal transfer learning)	Initial learning involved, and selection using *active learning*.

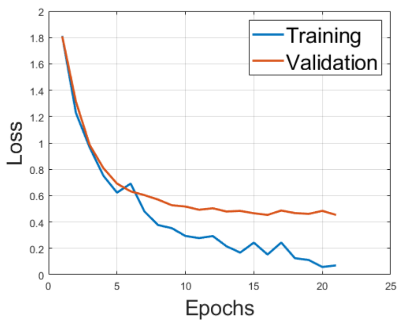
4.2.2 Results of the "RS" case

For other limitations (5%, 10%, 30%, 40%, or even 100% of the training set), the results are summarized and discussed later. The case of 100% means that we can utilize the totality of the training data without any limitation. Note that the datasets (SNPHEp-2, 13A, LSHEp-2) all exist in a labeled form. The labeling limitations suggested in this work are indicative of the potential afforded by active learning and are used here in order to demonstrate its effectiveness.

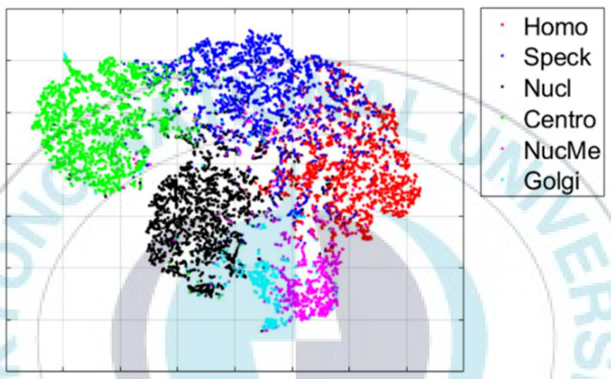
As said before, all of the following results concern the case of 20% of limitation. Fig. 36 shows the detailed results of the "**RS**" case. This case just consists of selecting randomly the 20% of images and use them to train the deep networks. Fig. 36 (a) shows the accuracy evolution over the training and validation sets (21 epochs). Fig. 36 (b) shows the loss evolution for the two sets. Fig. 36 (c) shows the visualization of the high-level features learned by the deep networks. All the visualizations here are obtained using the t-SNE [7]. Finally, Fig. 36 (d) shows the confusion matrix of the classification over the validation set. In all the confusion matrices shown here, "Homo", "Speck", "Nucl", "Centro", "NucMe", and "Golgi" refer to the Homogeneous, Speckled, Nucleolar, Centromere, Nuclear membrane and Golgi, respectively.



(**a**)



(**b**)



(**c**)

		Target Class					
		Homo	**Speck**	**Nucl**	**Centro**	**NucMe**	**Golgi**
Output Class	**Homo**	**89.69**	14.50	2.88	0	7.49	1.07
	Speck	8.13	**73.22**	1.51	1.41	8.23	2.93
	Nucl	0.91	4.43	**90.38**	0.55	3.44	9.60
	Centro	0.35	7.46	3.39	**98.00**	1.47	2.93
	NucMe	0.61	0.30	1.51	0	**46.68**	81.87
	Golgi	0.31	0.09	0.33	0.04	32.69	**1.60**

(**d**)

Fig. 36. Classification results of the "**RS**" case: (**a**) accuracy, (**b**) loss, (**c**) features' visualization, and (**d**) confusion matrix.

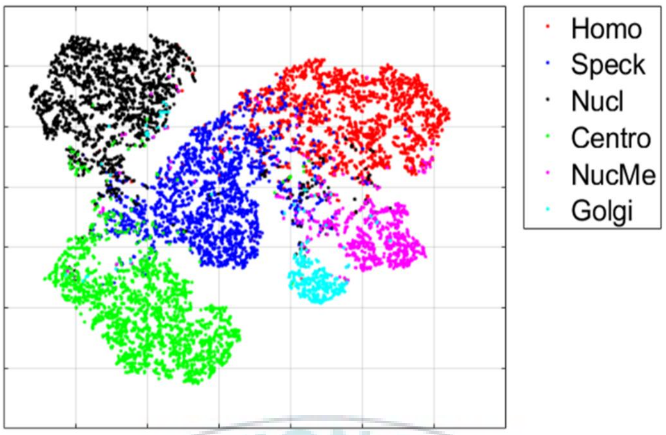
By analyzing the results, we can see that selecting randomly the data does not help for the generalization over the validation set. Two main observations can be highlighted from these results. First, there is a clear difference between the mean class accuracy (MCA) and the average classification accuracy (ACA). The MCA is 66.59% and the ACA is 81.13%. The ACA, which computes the overall accuracy by diving the number of correctly classified data by the total number of data, appears to take advantage of some of the classes that are very well discriminated. In particular, the Nucleolar (90.38%) and the Centromere (98.00%) contribute highly to establish the ACA in a very pleasant level.

On the other hand, the MCA, which computes the mean of all the classes' accuracies, is hugely impacted by the poor classification accuracy of the Golgi and Nuclear membrane cells. As part of the second main observation, as we can also remark in Fig. 36 (c) by analyzing the visualization of the features, there is an extreme confusion between the two cells' clusters (Golgi in magenta and Nuclear membrane in cyan). In fact, only 1.6% of the Golgi are well classified, while 81.87% of them are misclassified as Nuclear membrane, as we can see in the confusion matrix depicted in Fig. 36 (d). Also, only 46.68% of the Nuclear membrane are well classified. The two cellular types are certainly the most complicated to discriminate.

The first reason is that both types are always under-represented among the available data in all the existing datasets. There are only 375 Golgi and 814 Nuclear membrane instances in the training set, while all the others cell types contain each at least 2,100 images. The second reason is the complexity of their shape. Having the possibility of using only 20% of the training set, which diminishes again their number among the selected data for training, contributes to make the discrimination harder for the two cells. This pointed fact represents the principal observation of the present work. While having a limited number of labeled data in hands, the classification of these two cell types (Golgi and Nuclear membrane) becomes really complex.

4.2.3 Results of the "AL" case

Fig. 37 shows the results for the second case ("AL"). As for the first one, this case consists of not using the initial learning but, on the contrary, selects the 20% of data with active learning. The MCA for this case is 90.35% and the ACA is 91.51%. As we can notice in the confusion matrix in Fig. 37 (b), most of the cells maintain a quite fair classification result. And even more importantly, the huge confusion between the Golgi (we have 87.47% of accuracy) and Nuclear membrane (85.50%) has clearly diminished. The visualization in Fig. 37 (a) shows a noticeable separation of the two clusters compared with Fig. 36 (c). We can notice, in these results, the improvement afforded by the active learning-based selection. By selecting, precisely for annotation, the data for which the networks are the most confused about, active learning decreases the discrimination's complexity of the most difficult cells. At the same time, it maintains a good accuracy for the others cellular types.



(**a**)

Output Class	Target Class					
	Homo	**Speck**	**Nucl**	**Centro**	**NucMe**	**Golgi**
Homo	**89.56**	4.90	1.56	0.23	5.77	0.53
Speck	8.83	**91.09**	3.30	3.00	4.55	2.13
Nucl	0.87	1.41	**93.26**	1.36	2.09	5.07
Centro	0.22	1.75	0.66	**95.22**	0.37	1.33
NucMe	0.39	0.72	0.75	0.05	**85.50**	3.47
Golgi	0.13	0.13	0.47	0.14	1.72	**87.47**

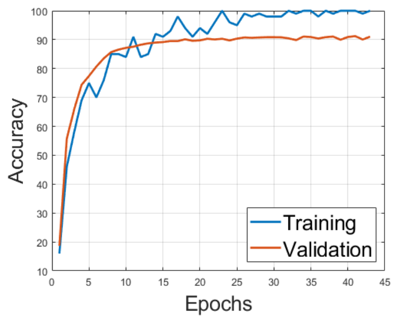
(**b**)

Fig. 37. Classification results of the “**AL**” case: (**a**) features’ visualization, and (**b**) confusion matrix.

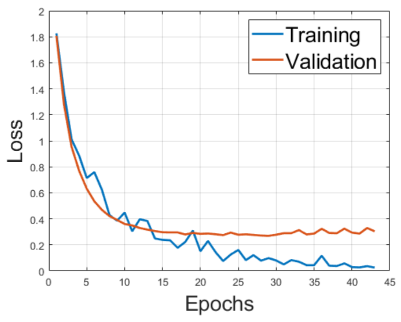
4.2.4 Results of the “IN-RS” case

From here, we discuss the results obtained when an initial learning is performed in order to build the pre-trained model. The first case (“**IN-RS**”) consists of selecting the 20% randomly in order to perform fine-tuning. In Fig. 38 (a)-(d), we have, respectively, the accuracy, loss, visualization of the features and confusion matrix for the “**IN-RS**” case.

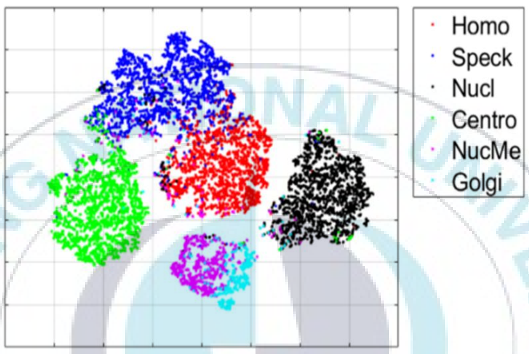
Here also (concerning the results in Fig. 38), we can notice how the extreme confusion remains present even after we apply cross-modal transfer learning. The MCA is 72.08%, which is better than the “**RS**” case. However, the poor accuracy (2.56%) of the Golgi really pulls down the MCA, even though the other cells accomplish excellent accuracies (55.86% for the Nuclear membrane). The ACA is 87.54% for this case. Note that the initial learning process increases the overall accuracy (the ACA goes from 81.13% to 87.54% between “**RS**” and “**IN-RS**”), but not for the two most difficult cellular types.



(**a**)



(**b**)



(**c**)

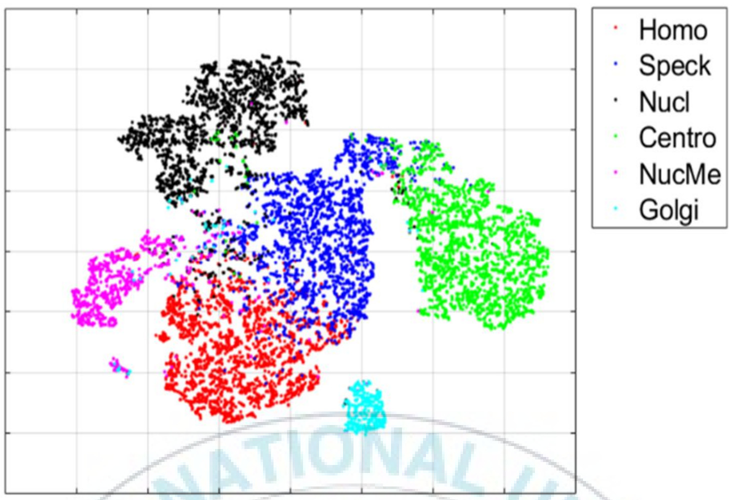
Output Class	Target Class					
	Homo	**Speck**	**Nucl**	**Centro**	**NucMe**	**Golgi**
Homo	**90.23**	5.22	0.15	0	6.68	2.29
Speck	5.19	**89.93**	1.71	2.56	3.63	1.14
Nucl	0.79	0.79	**96.79**	0.28	1.26	10.96
Centro	0.39	1.55	0.24	**97.16**	0.03	2.04
NucMe	0.56	0.12	0.07	0	**55.86**	81.01
Golgi	2.77	3.39	1.04	0	32.54	**2.56**

(**d**)

Fig. 38. Classification results of the "**IN-RS**" case: (**a**) accuracy, (**b**) loss, (**c**) features' visualization, and (**d**) confusion matrix.

4.2.5 Results of the “IN-AL” case

The second case, denoted as “**IN-AL**”, consists of selecting the 20% with active learning in order to perform fine-tuning. In Fig. 39 (a) and Fig. 39 (b), we have, respectively, the visualization of the features and the confusion matrix for the “**IN-AL**” case. As for the “**AL**” results discussed previously, we can notice how active learning permits to tackle the extreme confusion between the Golgi and the Nuclear membrane. This can be noticed in Fig. 39 (a) with the two clusters being completely disjoint, and in Fig. 39 (b), where we see that the two cells accomplish reasonable accuracy. For this “**IN-AL**” case, the MCA is 91.76% and the ACA is 92.77%. The most important observation here is that active learning strongly minimizes the divergence between the two metrics by assuring a quite fair discrimination for all of the cellular types. Note that all of these results are obtained by using only 20% of the available training instances in the large HEp-2 dataset.



(**a**)

		Target Class					
		Homo	**Speck**	**Nucl**	**Centro**	**NucMe**	**Golgi**
Output Class	**Homo**	**92.04**	5.37	1.84	0.09	5.53	0.53
	Speck	6.61	**90.50**	1.89	1.68	2.58	2.13
	Nucl	0.87	1.36	**94.01**	1.14	1.72	5.07
	Centro	0.13	2.05	1.04	**97.04**	0.36	1.33
	NucMe	0.26	0.68	0.89	0	**88.70**	2.67
	Golgi	0.09	0.04	0.33	0.05	1.11	**88.27**

(**b**)

Fig. 39. Classification results of the "**IN-AL**" case: (**a**) features' visualization, and (**b**) confusion matrix.

In Fig. 40, we show the classification accuracies of the three most difficult cells for the four cases ("**RS**", "**AL**", "**IN-RS**", "**IN-AL**"). We can notice how both cases that use active learning, by allowing to prioritize the annotation of the most difficult cells, "correct" the classification accuracies of the cases without active learning (especially for the Golgi and Nuclear membrane).

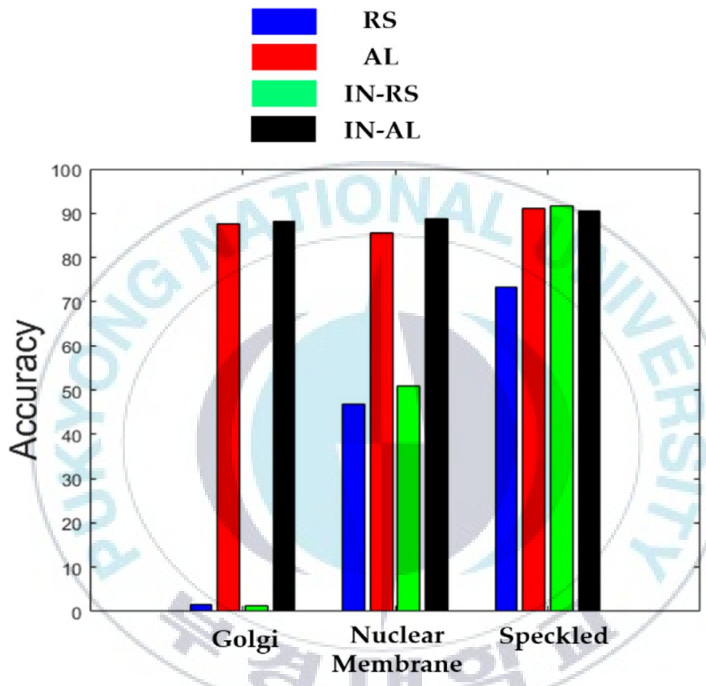


Fig. 40. Classification accuracy of the three most difficult cellular patterns (the Golgi, the Nuclear membrane and the Speckled).

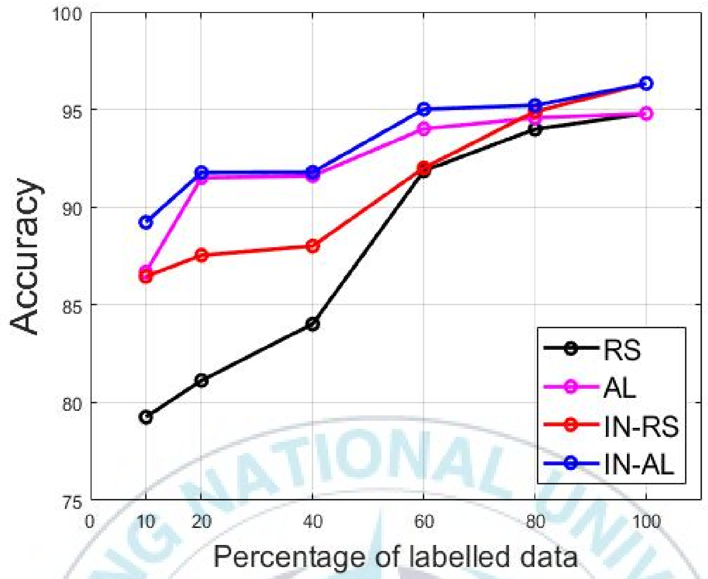


Fig. 41. Evolution of the accuracy with other limitations in terms of labeling. The accuracy is shown for the 4 different cases discussed above.

In Fig. 41, we show the summary of the results (ACA) for the others limitations. We show for the 10%, 20% (discussed in details previously), 40%, 60%, 80% and 100%. As explained before, the 100% case refers to the fact of using all the available training data. There is no active learning process in this case since all the data are supposed to be labeled. In this case, and only for this case, random sampling and active learning results are the same, as we can notice in Fig. 41.

For all the other limitations, we can see how active learning can help to achieve satisfying results even though we do not have access to the totality of the training data. In fact, for all the limitations, active learning-based labeling provides accuracies that are superior to 90%. It is only for the case of 10% limitation that active learning without cross-modal transfer learning ("**AL**") achieves 86.68% (see Fig. 41). However, this result can be significantly improved by using cross-modal transfer learning, as proposed in this work. In that case ("**IN-AL**"), the accuracy for the 10% limitation reaches 89.23%. In other words, active learning coupled with cross-modal transfer learning allows to achieve satisfying discrimination results even with a few number of labeled data in hands.

In order to show the contribution of cross-modal transfer learning, we show in parallel the classification accuracy when there is no limitation (100% of training data available) for the case where no initial learning is performed and for the case where we use the small dataset in order to build the pre-trained model and perform cross-modal transfer learning. Fig. 42 shows the comparison. We can remark how using the initial learning really improves the overall accuracy of the networks.

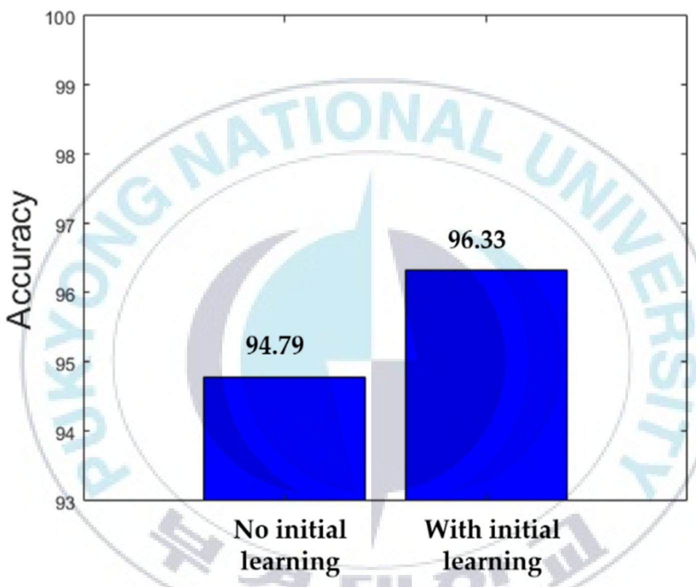


Fig. 42. Accuracy improvement with the initial learning in case of 100% of training data available.

In Table 7, we show the results of some different approaches. Note that most of the approaches have been proposed for either the ICPR2012 or 13A datasets. In this comparative study, we tried to see how these approaches react on the bigger and more complex LSHEp-2 dataset. Note also that none of these approaches utilize active learning. The comparisons are done with the use of the totality of the training data and the aim of the comparative study is more to demonstrate the effectiveness of our deep parallel networks and the contribution of the cross-modal transfer learning process.

Table 7. Comparative study using the LSHEp-2 dataset.

Methods	**Accuracy (ACA)**
Handcrafted features-based approach [114]	86.61%
LeNet-5-like CNN without transfer learning [80]	88.75%
VGG-16-like network without transfer learning [85]	90.23%
Transfer learning using the pre-trained VGG-19	91.57%
Transfer learning using the pre-trained AlexNet	92.41%
Transfer learning using the pre-trained VGG-16 [94]	92.89%
CNN with a Deep Residual Module [89]	94.15%
Transfer learning using the pre-trained ResNet-50	94.36%
Our proposed deep parallel residual nets without cross-modal transfer learning	94.79%
Cross-modal transfer learning using ResNet-50 [100]	95.94%
Our proposed deep parallel residual nets with cross-modal transfer learning	96.33%

Similar hyperparameters' settings were used for the training procedures of the models used here for comparison. The learning rate was set to 0.001 and training is terminated after the loss plateaus for 10 consecutive epochs. We have used a mini-batch of 128. Except in case of our method, for all the methods involving transfer learning, the parameters were updated for all the layers, since all of the pre-trained models were previously trained on ImageNet. Data augmentation, using the same technique as previously explained, was applied for the two methods that do not involve transfer learning, since the models are trained from the scratch.

As we can see in Table 7, most of the approaches using the models that were pre-trained on ImageNet perform less than the ones that use cross-modal transfer learning.

The state-of-the-art method in [100] utilize ResNet-50 but with an initial learning performed by using the ICPR2012 dataset, while their targeted dataset is the 13A. Our proposed method uses the deep parallel networks and the SNPHEp-2 dataset was utilized for the initial learning. Another state-of-the-art method is the DRC-Net [89], which achieves 94.15% on the LSHEP-2 dataset. We can notice that the accuracies shown in Fig. 41 are similar with these state-of-the-art performances. Active learning coupled with cross-modal transfer learning allows to achieve pleasant performance even with limitations in terms of labeling.

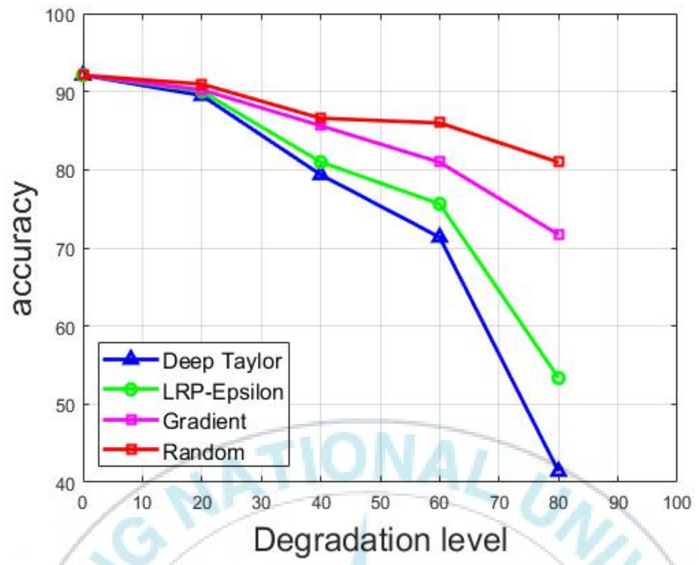
Note that models like VGG-16, VGG-19 and AlexNet require a substantial memory because of the enormous number of parameters generated by the fully connected layers at the end of the network. The residual networks (our parallel networks and the ResNet-50) also require a lot of memory but the computational complexity is far less compared to the other networks. This is explained by the global averaging pooling layer which efficiently minimizes the computational complexity by diminishing the total number of parameters of the networks. Another important point to note is that all the methods used in the comparative study in Table 7 necessitate only one single training procedure. On the other hand, every case that involves active learning in our method necessitates several training procedures because of the iterative labeling process. This fact can be considered as the principal limitation of our method, as the cascade of training procedures elongate the time needed to build the final model. But, as previously discussed, our aim is to demonstrate that we can achieve quite pleasant performance with only a limited number of labeled data in hands.

4.2.6 Results of the explainable active learning

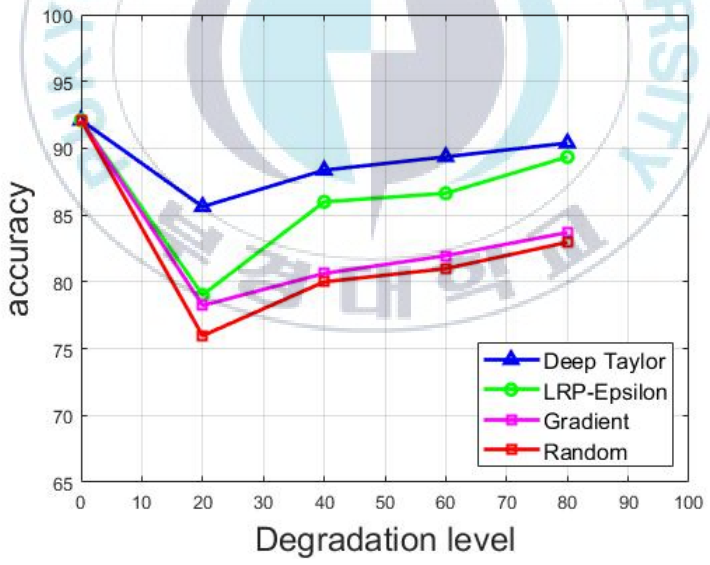
In this part, we present the results obtained by using the proposed explainable active learning, which utilizes an XAI-based selection method. As mentioned before, we start by selecting the best relevance map method for our data. One way to evaluate a relevance map, as explained in details in section 2.3.3, is to see how the network reacts when we remove the most relevant parts of the image. These relevant parts are, of course, generated by the relevance map. One way of evaluation is to remove the most important parts of the data and retrain the network. If the accuracy decreases, it means that the relevance map was right by showing the most important parts of the data for the network. In case the accuracy does not decrease notably, it means that the relevance map has failed to reveal the most relevant parts of the data.

Fig. 43 shows the results of the different experiments conducted for that purpose. In Fig. 43 (a) we have the results of the ROAR experiment. As previously explained (in section 2.3.3), ROAR consists of removing the relevant features, retrain and re-test the network using the modified dataset. Here, as also performed by Hooker et al. [37], the removal consists of replacing the relevant pixels by an average value computed using all the pixel values of the image.

In Fig 43 (a), the degradation level denotes the amount, expressed in percentage, of the pixels that were modified. The beginning denotes 0% of degradation, a situation where none of the pixels were changed, and the end shows 80% of degradation, meaning that 80% of the pixels (following their relevance scores) were modified. In case of the ROAR setup, the performance of the model is expected to decrease if the relevance map really shows the most important features. A bigger drop in the accuracy shows a better relevance map method while a lower decrease shows an inefficient relevance map. “Random” denotes the fact of removing the pixels randomly. An efficient relevance map method is expected to produce much more accuracy’s reduction than a random removal.



(a)



(b)

Fig. 43. Evaluation of the different relevance maps methods: (a) using the **ROAR** and (b) using the **KAR**.

DeepTaylor and LRP methods have provided the best relevance maps for our data. The term “LRP-Epsilon” in Fig. 43 refers to a more efficient version of the LRP algorithm [36]. As we can notice in Fig. 43 (a), the accuracy of the network degrades significantly when we remove the parts (the pixels) that are selected as important by these two methods. They perform better than the gradients-based relevance maps and a random removal. “Random removal” consists of removing the pixels randomly, without any coherence, as opposed to the removal because the pixels were designated as relevant by a certain method.

Fig 43 (b) sows the results of the KAR experiments. KAR is the inverse of ROAR: we remove the irrelevant pixels and keep the relevant ones. The accuracy is expected to not vary too much. Also, as for the ROAR, DeepTaylor maps have produced the most consistent relevance maps for our data and networks, as we can notice in Fig. 43 (b). That is the reason why DeepTaylor maps were preferably selected in order to perform the proposed explainable active learning scheme. We recall here again that the maps are produced by using our deep parallel residual networks presented before.

Fig. 44 shows the results obtained on the testing set of the LSHEP-2 dataset by using only 20% of the training data with our explainable active learning method. The network achieves 92.77%. In comparison, using the totality (100% of data) of the training data gives 96.33% of accuracy on the test set. Even though the results obtained by utilizing the totality of the data are better, the results demonstrate that active learning can ensure a quite pleasant performance while alleviating the labeling process.

Fig. 45 shows the comparison between the random selection (RS), the conventional selection, denoted as “CAL”, and the proposed selection, denoted as XAL, for explainable active learning. We can see that the proposed selection method increases the performance of the classifiers. In addition, the proposed selection method has the advantage of being explainable. We can justify the reason why certain images are selected by using their relevance maps. In that manner, the oracle (human annotator) can clearly understand the reasons why the images are being selected for the labeling process.

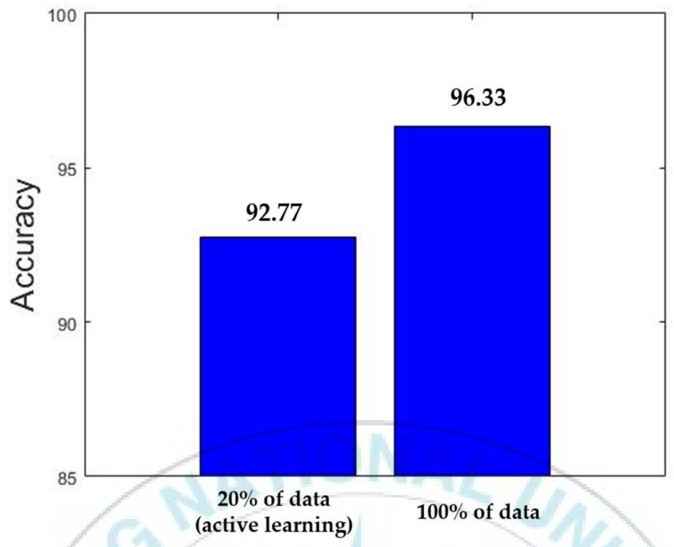


Fig. 44. Results on the LSHEp-2 dataset: with only 20% of the data (selected using the proposed explainable active learning), compared to using 100% of data.

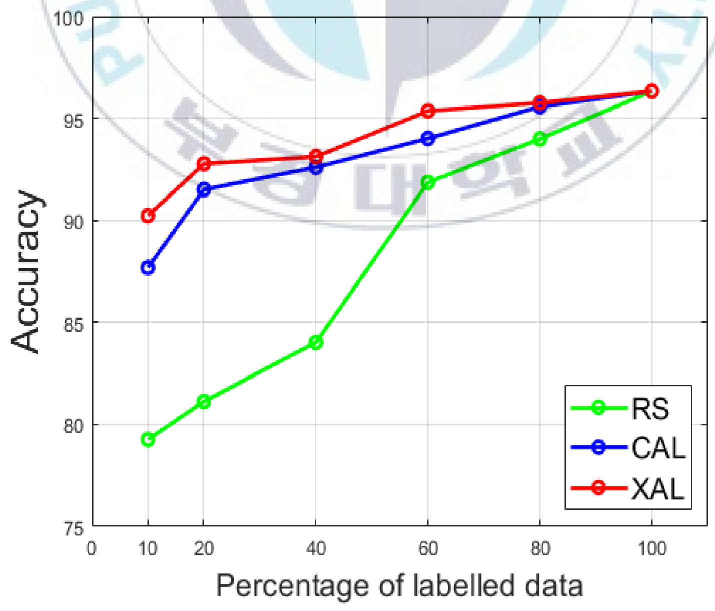


Fig. 45. Comparison between the random sampling (RS), conventional active learning (CAL) and the proposed explainable active learning (XAL).

In fact, by analyzing the relevance maps of certain images, we can clearly understand why they have been marked as “ambiguous”, or “difficult”, or even “uncommon” by the XAL selection process. In the following figures, we show and discuss about some examples of the data (and their relevance maps) that were selected for the annotation by our proposed explainable active learning.

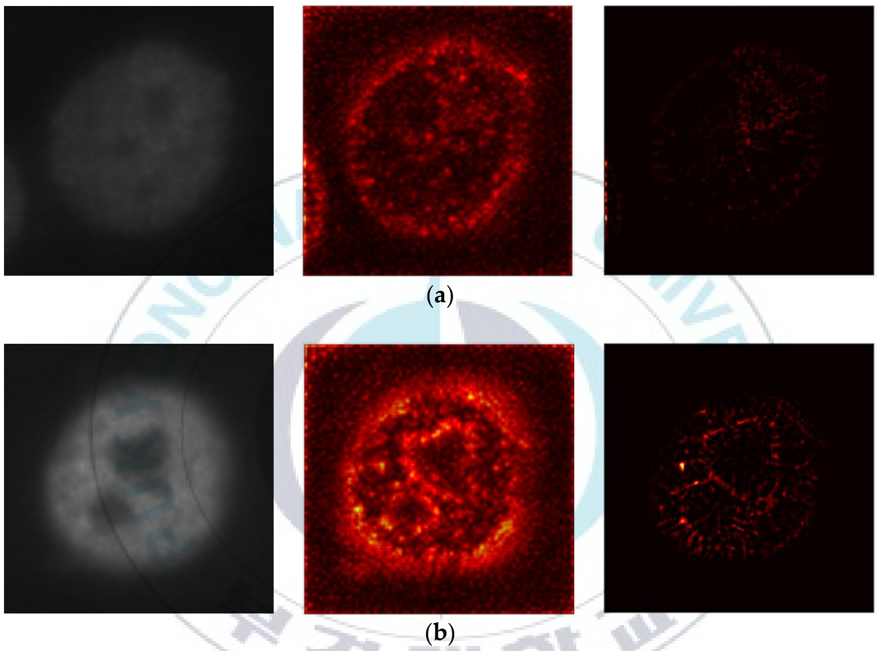
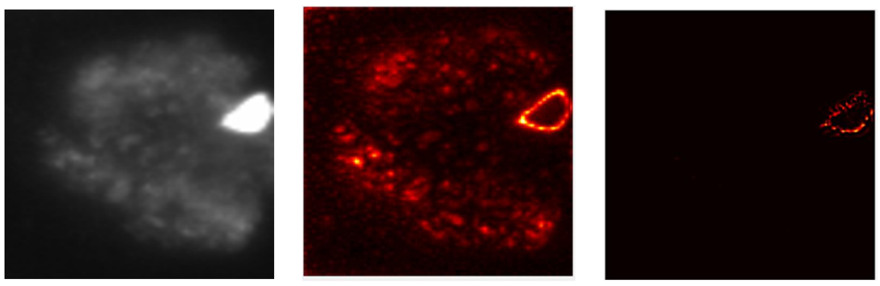
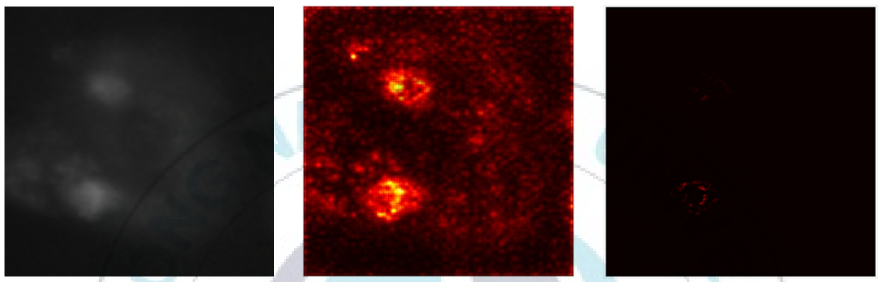


Fig. 46. **Homogeneous** images with their corresponding maps: (**a**) Homogeneous pattern and its relevance maps; (**b**) another Homogeneous pattern and its relevance maps.

In Fig. 46, we show some examples of cellular images and their corresponding relevance maps. In Fig. 46 (a), we have an example of a homogeneous cellular pattern and its corresponding relevance maps. In Fig. 46 (b), we have another homogeneous pattern and its corresponding relevance maps. For all the images depicting the relevance maps, the second column shows the DeepTaylor map while the third column depicts the LRP map. We can see from the images in Fig. 46 that the relevance maps capture some specific characteristics of the homogeneous class. The classifier had shown a quite high confidence in the discrimination of these images, showing probability values higher than 95% for their corresponding class.


(a)


(b)

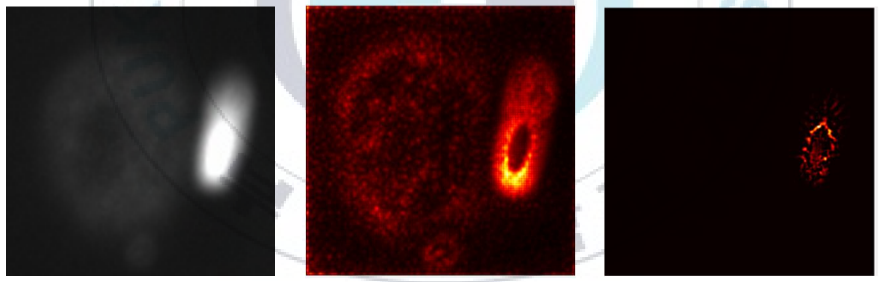

(c)

Fig. 47. Example of some homogeneous images, with their corresponding maps, that were selected by the XAL.

In Fig. 47, we show some of the images belonging to the homogeneous class that were selected for the annotation. By comparing the relevance maps of the images in Fig. 46 with the ones shown in Fig. 47, we can remark how their relevance maps display quite different characteristics. Especially the LRP maps. All the images in Fig. 47 were designated as ambiguous as their relevance maps are quite atypical. In fact, the probabilities outputted by the classifier concerning these images also show a strong uncertainty. The probability values corresponding to the homogeneous class were only about 57%, 46% and 44% for the images in Fig. 47 (a), (b) and (c), respectively. This situation demonstrates that the ambiguity in the relevance map is correlated with the ambiguity in the classifier's output.

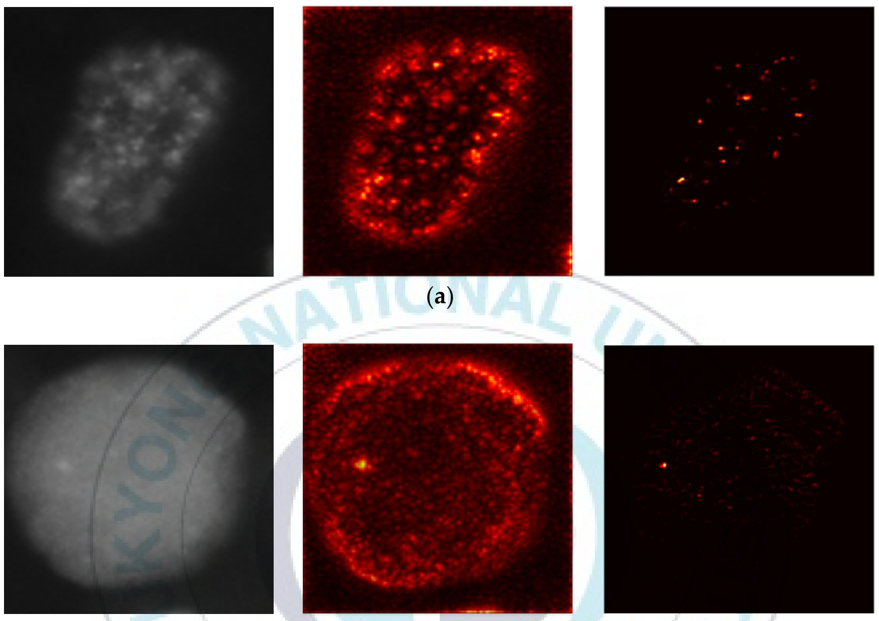


Fig. 48. **Speckled** images with their corresponding maps: (**a**) Speckled pattern and its relevance maps; (**b**) another Speckled pattern and its relevance maps.

Fig. 48 (a) shows an example image of a common pattern of the speckled type. For this image, the probability for the right class was more than 99%. An interesting example is shown in Fig. 48 (b). The classifier was confident at 99% that this image belongs to the homogeneous class. We can find many images like the one depicted in Fig. 48 (b) in the speckled class. That is the reason why this class is one of the most difficult classes (as discussed previously - we can see this fact in Fig. 40).

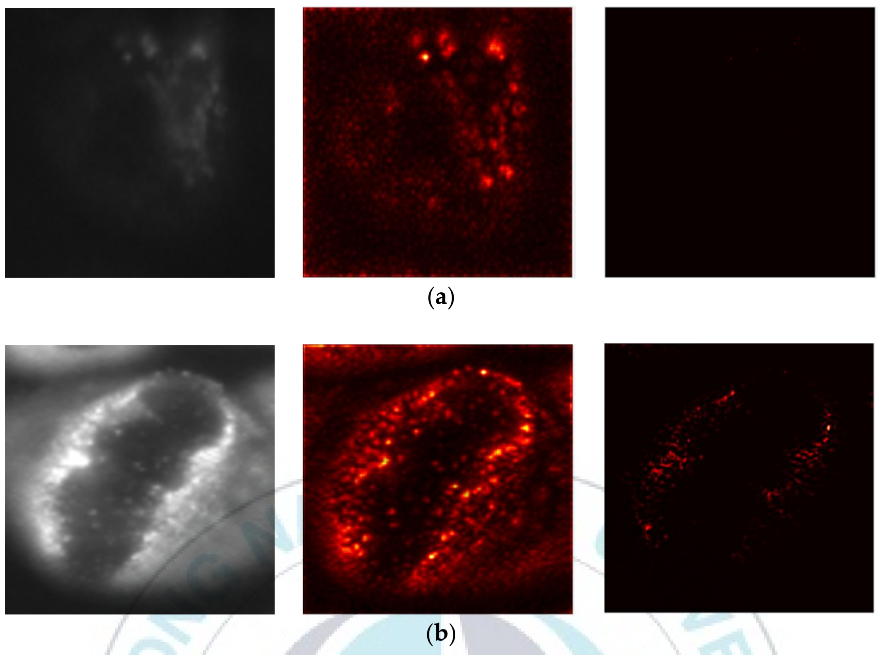


Fig. 49. Example of some speckled images, with their corresponding maps, that were selected by the XAL.

Fig. 49 shows examples of some of the speckled images that were selected for annotation. In comparison, we can denote the extreme differences in the relevance maps that exist between the two groups of images (Fig. 48 and Fig. 49). Speckled images depicted in Fig. 49 (a) and (b) are atypical, that's why they were selected by the XAL.

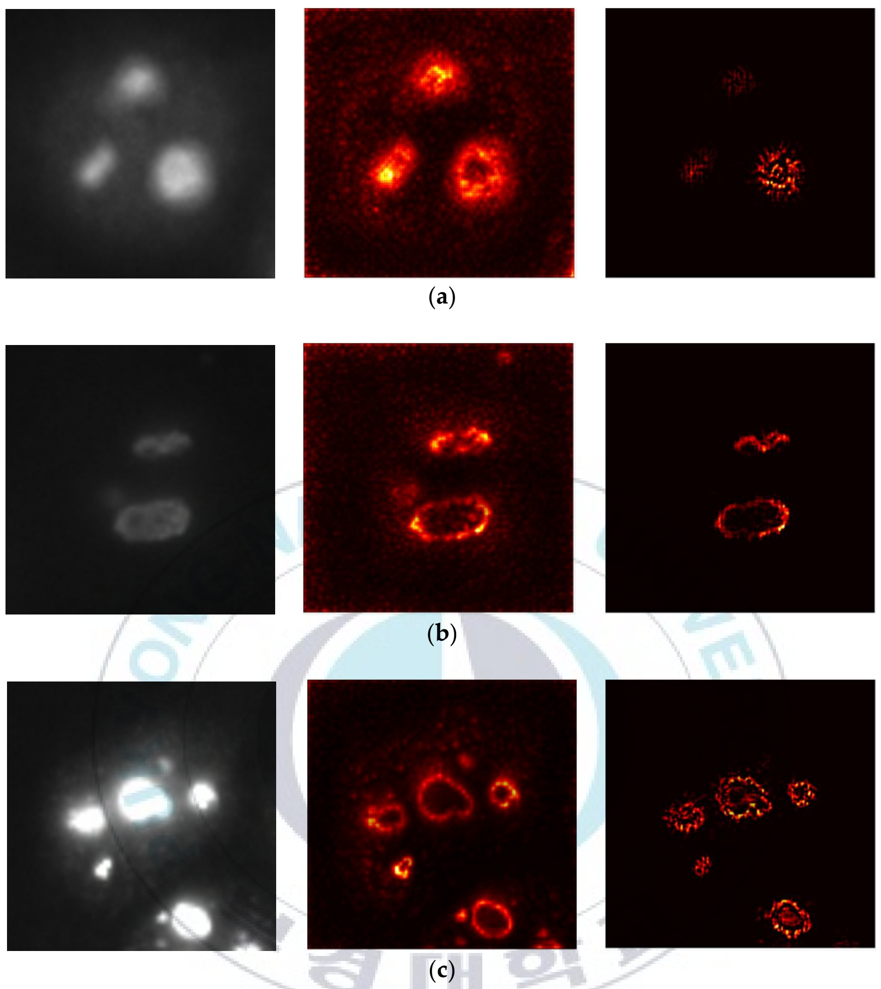


Fig. 50. **Nucleolar** images with their corresponding maps: (**a**) Nucleolar pattern and its relevance maps; (**b**) another Nucleolar pattern and its relevance maps.

In Fig. 50 (a) - (c), we can see some examples of the nucleolar class for which the classifier was highly confident. We can notice how their respective relevance maps share some specific characteristics of the nucleolar category.

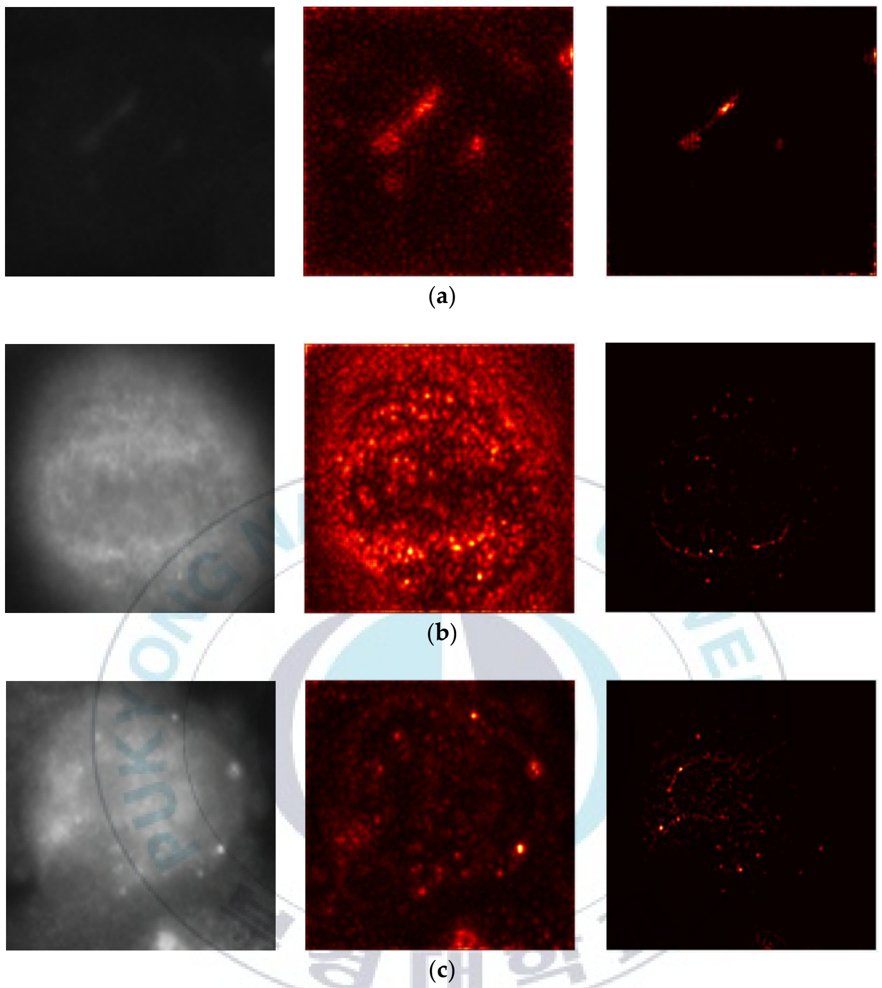
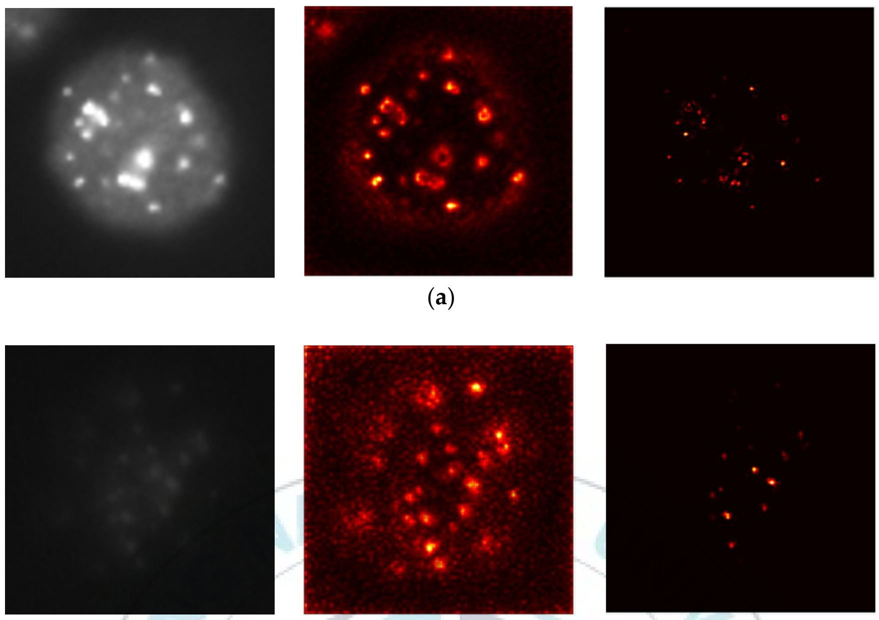


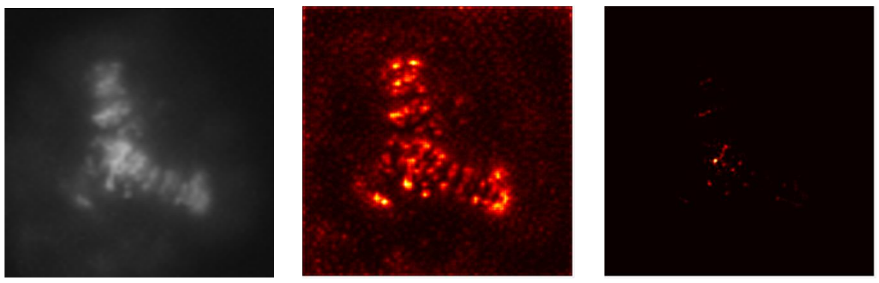
Fig. 51. Example of some nucleolar images, with their corresponding maps, that were selected by the XAL.

On the contrary, Fig. 51 shows some images that are ambiguous for the nucleolar class. For the images in Fig. 51 (a) and (b), the classifier was relatively confident, giving around 70% of probability for both images. Their relevance maps give an explanation of why these images were selected for the annotation. We can see how their respective relevance maps differ totally with the ones shown in Fig. 50. In Fig. 51 (c), we show an example for which the confidence was high but for a wrong class. This image was classified with 96% of probability as a speckled image.

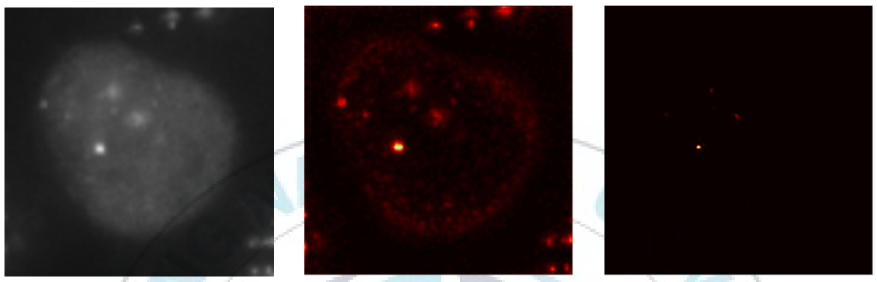

(**a**)

(**b**)

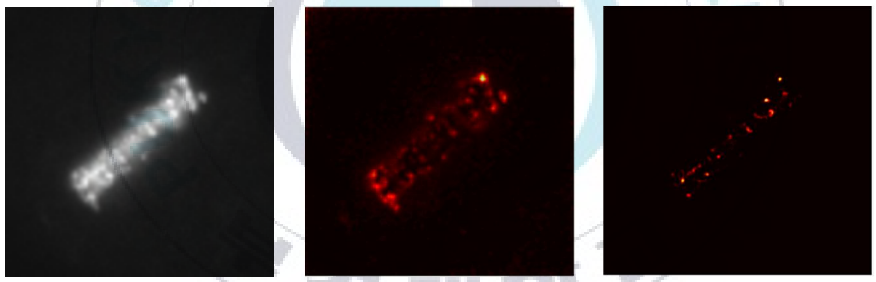
Fig. 52. **Centromere** images with their corresponding maps: (**a**) Centromere pattern and its relevance maps; (**b**) another Centromere pattern and its relevance maps.



(a)



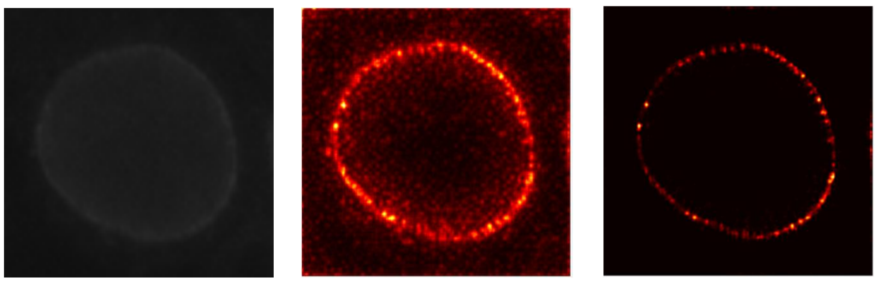
(b)

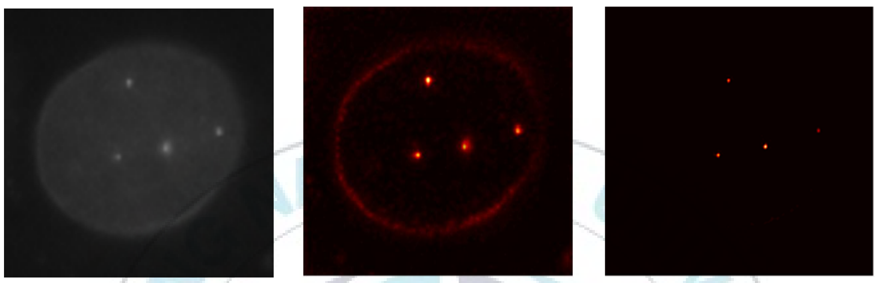


(c)

Fig. 53. Example of some centromere images, with their corresponding maps, that were selected by the XAL.

In Fig. 52 (a) and (b), we have examples from the centromere class for which the classifier was highly confident (more than 95% of probability for both images). On the contrary, the images depicted in Fig. 53 (a) – (c) denote the ambiguous images that were selected for the annotation. Here also, we can remark the notable differences between the relevance maps between the two groups of images (Fig. 52 and Fig. 53). For the three images portrayed Fig. 53, the probability values were around 55% for their corresponding class.


(a)


(b)

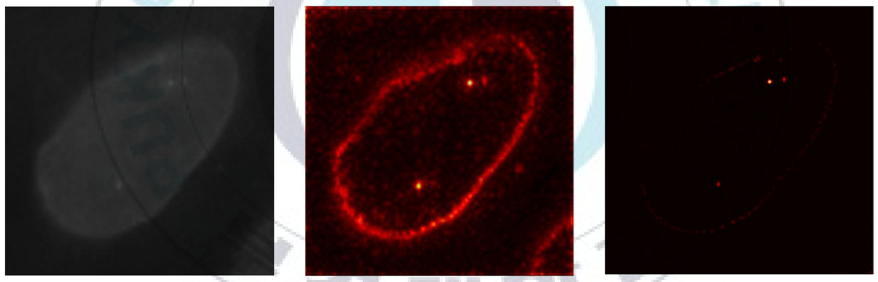
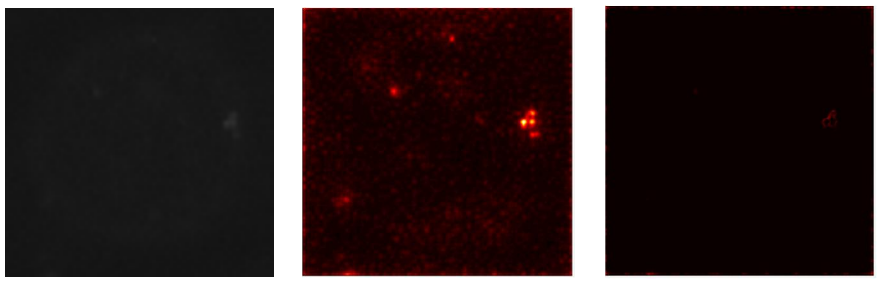
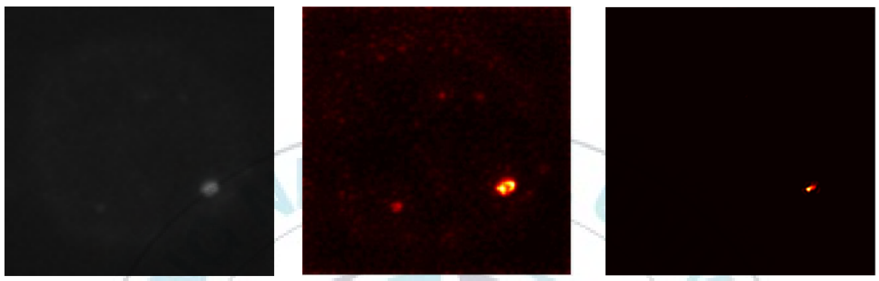

(c)

Fig. 54. **Nuclear membrane** images with their corresponding maps: (**a**) Nuclear membrane and its relevance maps; (**b**) another Nuclear membrane pattern and its relevance maps.

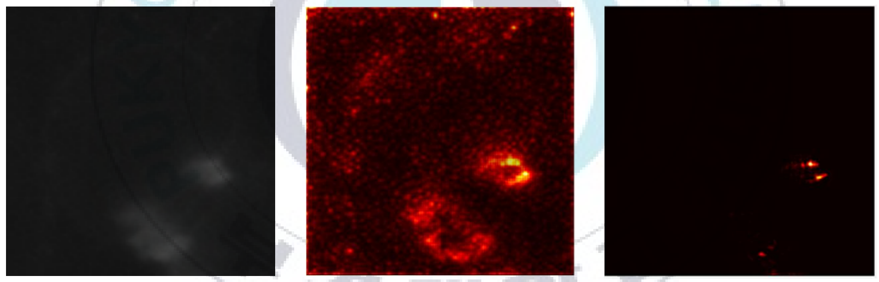
Fig. 54 (a) – (c) show the images from the nuclear membrane class for which the classifier was highly confident. We can remark how the typical features of this class are notably exposed by their relevance maps (especially the DeepTaylor maps in the second column). All the images that display these particular features are classified with a high confidence (more than 98% for the majority of them) by the classifier.



(a)



(b)



(c)

Fig. 55. Example of some nuclear membrane images, with their corresponding maps, that were selected by the XAL.

In Fig. 55 (a) – (c), we show the nuclear membrane images that were selected for the annotation. Notice how these images in Fig. 55 exhibit strongly atypical features for this particular class. By comparing the relevance maps shown in both Fig. 54 and Fig. 55, we can easily understand why the images in Fig. 55 were selected for the annotation. The probabilities outputted by the classifier for these images were all around 51%, which denotes a quite strong uncertainty. We can see how our proposed XAL method really incorporates the justification of the selection process.

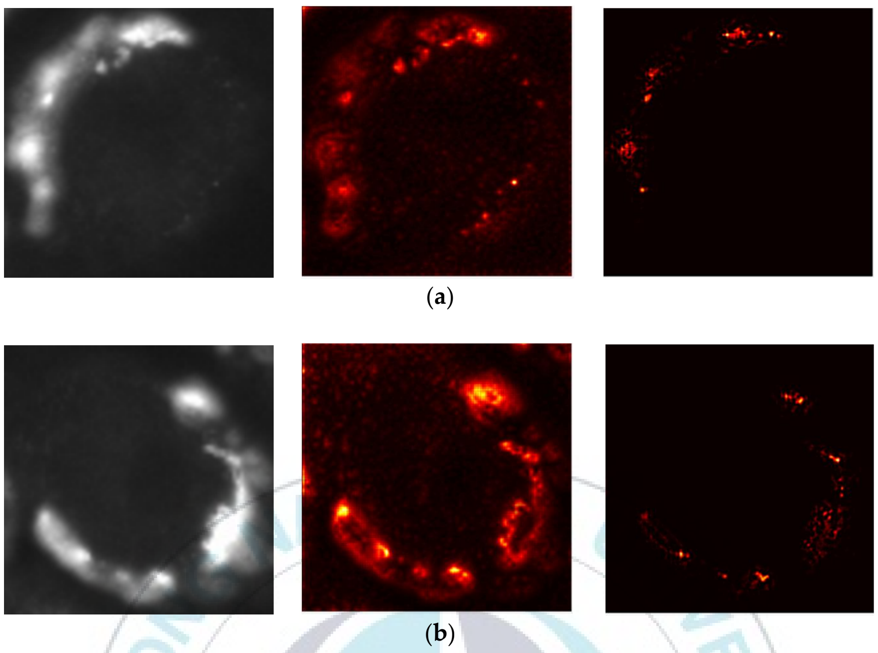


Fig. 56. **Golgi** images with their corresponding maps: (**a**) Golgi pattern and its relevance maps; (**b**) another Golgi pattern and its relevance maps.

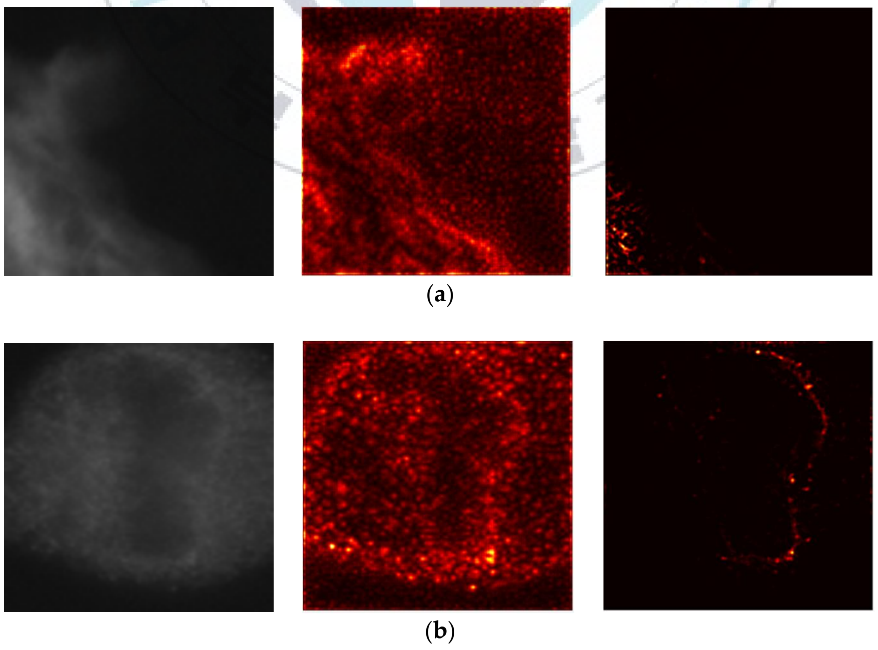


Fig. 57. Example of some Golgi images, with their corresponding maps, that were selected by the XAL.

In Fig. 56 (a) – (b), we show some examples from the Golgi class for which the classifier was highly confident. Their respective relevance maps demonstrate some specific features for this class. For these images, the probability was near 99% for their class. On the contrary, in Fig. 57 (a) – (b), we show the images from the same class that were selected for the annotation. Note how the relevance maps (especially the DeepTaylor maps) of the images shown in Fig. 57 do not share any consistent similarities with the ones shown in Fig. 56. For the two ambiguous images in Fig. 57, the probability values for their class were only around 56%, which denotes a certain uncertainty for these images. From all those images, we demonstrate the "explicability" of our selection method.

Fig. 58 shows the comparison between one of the actual state-of-the-art active learning methods, the variational adversarial active learning (VAAL) proposed by Sinha et al. [115], and our proposed explainable active learning (XAL) by using the CIFAR-10 dataset. Since this method was not proposed for the cellular images, we did not apply it for the HEp-2 cell dataset. Note that the results shown here for the VAAL method were reported as they were shown in their works.

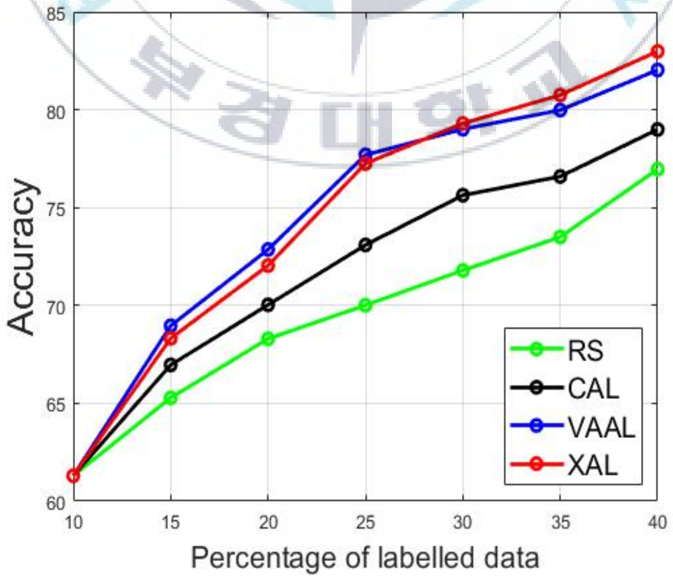


Fig. 58. Comparison using the CIFRA-10 dataset between the random sampling (RS), conventional active learning (CAL), the variational adversarial active learning (VAAL) and the proposed explainable active learning (XAL).

Here also, in Fig. 58, we can see that both active learning methods (VAAL and XAL) work better than the conventional selection way (CAL). VAAL and XAL work similarly in terms of performance. Note that our proposed selection method has the advantage of explainability, since the selection method proposed in the VAAL cannot be fully understood by human annotators. Their method principally relies on the output of the variational autoencoder. This network is also a black-box model whose decision has no interpretability at all. Hence, the labeling decision made by the VAAL remains also non interpretable. On the contrary, for our method, we did not utilize any model's output. All the selection method is based on the relevance maps computed by using the XAI methods.

V. CONCLUSIONS

The present work aimed to propose two methods for the classification of the HEp-2 cell images. The two methods have the particularity that they are specifically designed in order to really alleviate the labeling burden. Since most of the machine learning models utilize the supervised learning paradigm, and knowing that this approach requires the manual labeling of data, we think that the unsupervised learning approach is more than necessary. We then proposed two models that are based on different scenarios.

The first model is based on an unsupervised learning scenario where there is no need of labeled data in order to perform an end-to-end training of the model. On that purpose, we used deep clustering, a technique that combines conventional clustering methods with deep learning structures. A clustering layer was embedded in the middle of a DCAE in order to perform clustering on the latent space's features at every iteration of the training process. This technique allows the DCAE to produce features that are easily discriminable.

As a principal contribution, we proposed different techniques that can ensure a better reconstruction process. In the deep clustering literature, the reconstruction is performed only by the reconstruction loss of the DCAE. On the contrary, we have demonstrated that some techniques can be applied in order to improve the overall reconstruction quality of the DCAE. For that purpose, we proposed to use the pooling indices storage (by abandoning the use of the transposed convolution for the up-sampling process), the copy-and-concatenation technique and, more importantly, the attention-based network.

We showed in the results that, when the three techniques are all combined, the DCAE's reconstruction accuracy is efficiently maximized. Finally, for this first model, we presented a systematic analysis of the results in order to explore the influence of the reconstruction accuracy on the latent features.

Also, as another major contribution, we demonstrated that the reconstruction performance of the DCAE affects the quality of the learned latent features. Results were shown for different scenarios and the case-3 network, which utilizes all of the three techniques together, has provided the best reconstruction and clustering accuracies. The proposed case-3 network achieves 97.59% and 94.89% of discrimination accuracy in the SNPHEp-2 and the 13A datasets, respectively. This performance is far better than the results obtained by the conventional supervised handcrafted features-based methods. More importantly, this performance is similar with the actual deep learning-based state-of-the-art supervised learning methods in the literature of the HEp-2 cell classification. As we can remark, our method is unsupervised.

The second model is based on a semi-supervised learning scheme. We adopt the techniques of active learning in order to alleviate the data labeling process. As a primary contribution of this work, we have redefined active learning by proposing a data selection process that is based on XAI, called explainable active learning. Conventional methods utilize the outputs of the deep learning model for the selection process. We have proposed a method that completely abandons this output-based selection. Our method preferably utilizes the relevance maps in order to evaluate the confidence of the model. This selection method is task-agnostic, does not really depend on the model's performance and, most importantly, has the advantage of being explainable. The ability of being explainable is very crucial in the context of interpretability issues posed by the deep learning models.

In terms of the results, for this second model, we have first demonstrated that active learning, when coupled with a well-performed cross-modal transfer learning, can really provide better results for the discrimination of the HEp-2 cell images. We conducted detailed experiments in order to demonstrate the effectiveness of active learning using our dynamic networks. These networks were designed specifically in order to solve the huge complexity issues posed by the public HEp-2 cell image datasets. Results have shown their effectiveness both for the cross-modal transfer learning and active learning.

Secondly, the results demonstrated that our newly formulated XAI-based selection contributes to boost the performance of the model even with a quite limited number of labeled data (the method has achieved 92.77% of accuracy on the HEp-2 testing data with only 20% of the training data available). In comparison, using 100% of the training data achieves 96.33% of accuracy. Relevance maps based on DeepTaylor were utilized and allowed us to select quite informative data for the labeling process. Our newly proposed explainable active learning method performs at the same level with the actual state-of-the-art active learning method. However, our proposed selection method has the advantage of being explainable. And that is the most important contribution. Our other contribution papers can be found in [116-124].

VI. REFERENCES

[1] A. Rigon, P. Soda, D. Zennaro, G. Iannello, and A. Afeltra, "Indirect immunofluorescence in autoimmune diseases: Assessment of digital images for diagnostic purpose," *Cytometry Part B: Clinical Cytometry,* Vol. 72, No. 6, pp. 472–477, 2017.

[2] P. Foggia, G. Percannella, P. Soda, and M. Vento, "Benchmarking HEp-2 cells classification methods," *IEEE Transactions on Medical Imaging,* Vol. 32, pp. 1878–1889, 2013.

[3] C. Vununu, S.K. Lee, and K.R. Kwon, "A deep feature extraction method for HEp-2 image classification," *Electronics*, Vol. 8, 2018.

[4] A.K. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognition Letters*, Vol. 31, No. 8, pp. 651-666, 2010.

[5] A.K. Jain, M.N. Murty, and P.J. Flynn, "Data clustering: a review," *ACM Computing Survey*, Vol. 31, No. 3, pp. 264-323, 1999.

[6] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and Intelligent Laboratory Systems*, Vol. 2, pp. 37–52, 1987.

[7] L.V.D. Maaten and G.E. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research.*, Vol. 9, pp. 2579–2605, 2008.

[8] H. Bourlard and Y. Kamp, "Auto-association by multilayer perceptrons and singular value decomposition," *Biological Cybernetics,* Vol. 59, pp. 291–94, 1988.

[9] Y. Bengio, "Learning deep architecture for AI," *Foundations and Trends in Machine Learning*, Vol. 2, pp. 1-127, 2009.

[10] G.E. Hinton and R.R. Salakhutdinov, "Reducing the dimensionality of the data with neural networks," *Science*, Vol. 313, pp. 504-507, 2006.

[11] H.C. Shin, M.R. Orton, D.J. Collins, S.J. Doran, and M.O. Leach, "Stacked autoencoders for unsupervised feature learning and multiple organ detection in

a pilot study using 4D patient data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 35, No. 8, pp. 1930-1943, 2013.

[12] P. Vincent, H. Larochelle, and I. Lajoie, "Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion," *Journal of Machine Learning Research,* Vol. 11, pp. 3371-3408, 2010.

[13] A. Ng, "Sparse autoencoder lecture notes", 2013. Available online: https://web.stanford.edu/class/cs294a/sparseAutoencoder.pdf, Accessed on July 21, 2021.

[14] F. Li, H. Qiao, B. Zhang, and X. Xi, "Discriminatively boosted image clustering with fully convolutional auto-encoders," *arXiv preprint*, arXiv:1703.07980, 2017.

[15] X. Guo, X. Liu, E. Zhou, and J. Yin, "Deep clustering with convolutional autoencoders," In *Proceedings of the International Conference on Neural Information Processing (ICONIP)*, pp. 373–382, 2017.

[16] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, 2016.

[17] J. Yang, D. Parikh, and D. Batra, "Joint unsupervised learning of deep representations and image clusters," In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5147–5156, 2016.

[18] X. Peng, J. Feng, J. Lu, W.-Y. Yau, and Z. Yi, "Cascade subspace clustering," In *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI)*, pp. 2478–2484, 2017.

[19] E. Min, X. Guo, Q. Liu, G. Zhang, J. Cui, and J. Long, "A survey of clustering with deep learning: From the perspective of network architecture," *IEEE Access*, Vol. 6, pp. 39501–39514, 2018.

[20] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, “Deep clustering for unsupervised learning of visual features,” In *Proceedings of the 15th European Conference on Computer Vision (ECCV 2018)*, 2018.

[21] K.G. Dizaji, A. Herandi, and H. Huang, “Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization,” In *Proceedings of the IEEE International Conference on Computer Vision (ICCV 2017)*, pp. 5747–5756, 2017.

[22] A. Singh, S. Sengupta, and V. Lakshminarayanan, “Explainable deep learning models in medical image analysis,” *Journal of Imaging*, Vol. 6, No. 6, 2020.

[23] A.B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, et al., “Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI,” *Information Fusion*, Vol. 58, pp. 82–115, 2020.

[24] P. Zhu and M. Ogino, “Guideline-Based Additive Explanation for Computer-Aided Diagnosis of Lung Nodules,” In *Interpretability of Machine Intelligence in Medical Image Computing and Multimodal Learning for Clinical Decision Support*; Springer: Cham, Switzerland, pp. 39–47, 2019.

[25] H. Lee, S.T. Kim, and Y.M. Ro, “Generation of Multimodal Justification Using Visual Word Constraint Model for Explainable Computer-Aided Diagnosis,” In *Interpretability of Machine Intelligence in Medical Image Computing and Multimodal Learning for Clinical Decision Support*; Springer: Cham, Switzerland, pp. 21–29, 2019.

[26] M. Stano, W. Benesova, and L.S. Martak, “Explainable 3D convolutional neural network using GMM encoding,” In *Proceedings of the 12th International Conference on Machine Vision (ICMV 2019)*, 2019.

[27] M.D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” In *Proceedings of the European Conference on Computer Vision (ECCV 2014)*, pp. 818–833, 2014.

[28] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg, “Smoothgrad: Removing noise by adding noise,” *arXiv preprint*, arXiv:1706.03825, 2017.

[29] K. Simonyan, A. Vedaldi, and A. Zisserman, “Deep inside convolutional networks: Visualising image classification models and saliency maps,” *arXiv preprint*, arXiv:1312.6034, 2013.

[30] J.T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, “Striving for simplicity: The all convolutional net,” *arXiv preprint*, arXiv:1412.6806, 2014.

[31] A. Shrikumar, P. Greenside, A. Shcherbina, and A. Kundaje, “Not just a black box: Learning important features through propagating activation differences,” *arXiv preprint*, arXiv:1605.01713, 2016.

[32] M. Sundararajan, A. Taly, and Q. Yan, “Axiomatic attribution for deep networks,” In *Proceedings of the 34th International Conference on Machine Learning (ICML 2017)*, pp. 3319–3328, 2017.

[33] P.J. Kindermans, K.T. Schütt, M. Alber, K.R. Müller, D. Erhan, B. Kim, and S. Dähne, “Learning how to explain neural networks: Patternnet and patternattribution,” *arXiv preprint*, arXiv:1705. 05598, 2017.

[34] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.R. Müller, and W. Samek, “On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation,” *PLoS ONE*, Vol. 10, 2015.

[35] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K.R. Müller, “Explaining nonlinear classification decisions with deep Taylor decomposition,” *Pattern Recognition*, Vol. 65, pp. 211–222, 2017.

[36] W. Samek, A. Binder, G. Montavon, S. Bach, and K.R. Muller, “Evaluating the visualization of what a deep neural network has learned,” *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 28, No. 11, pp. 2660-2673, 2016.

[37] S. Hooker, D. Erhan, P.-J. Kindermans, and B. Kim, “A benchmark for interpretability methods in deep neural networks,” In *Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*, 2019.

[38] J. Adebayo, J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt, and B. Kim, "Sanity checks for saliency maps," In *Proceedings of the 32nd Conference on Neural Information Processing Systems (NeurIPS 2018)*, 2018.

[39] D.D. Lewis and J. Catlett, "Heterogeneous uncertainty sampling for supervised learning," In *Proceedings of the Eleventh International Conference on Machine Learning,* pp. 148–156, 1994.

[40] C.E. Shannon, "A mathematical theory of communication," *ACM SIGMOBILE Mobile Computing Communications Review*, Vol. 5, pp. 3–55, 2001.

[41] D.D. Lewis and W.A. Gale, "A sequential algorithm for training text classifiers," In *Proceedings of the 17th ACM International Conference on Research and Development in Information Retrieval*, pp. 3–12, 1994.

[42] T. Scheffer, C. Decomain, and S. Wrobel, "Active hidden markov models for information extraction," In *International Symposium on Intelligent Data Analysis*; Springer: Berlin/Heidelberg, Germany, pp. 309–318, 2001.

[43] S. Tong and D. Koller, "Support vector machine active learning with applications to text classification," *Journal of Machine Learning Research*, Vol. 2, pp. 45–66, 2001.

[44] N. Abe and H. Mamitsuka, "Query learning strategies using boosting and bagging," In *Proceedings of the Fifteenth International Conference on Machine Learning (ICML 1998)*, pp. 1-8, 1998.

[45] L.Z. Huo and P. Tang, "A batch-mode active learning algorithm using region-partitioning diversity for SVM classifier," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, Vol. 7, pp. 1036–1046, 2014.

[46] H.I. Suk and D. Shen, "Deep learning-based feature representation for AD/MCI classification," In *Proceedings of the Medical Image Computing and Computer-Assisted Intervention*. In *Lecture Notes in Computer Science*, Vol. 8150, pp. 583–590, 2013.

[47] H.I. Suk, S.W. Lee, and D. Shen, "Latent feature representation with stacked auto-encoder for AD/MCI diagnosis," *Brain Structure & Function*, Vol. 220, No. 2, pp. 841-59, 2013.

[48] A. Payan and G. Montana, "Predicting Alzheimer's disease: a neuroimaging study with 3D convolutional neural networks," *arXiv preprint*, arXiv:1502.02506, 2015.

[49] H.I. Suk, C.Y. Wee, S.W. Lee, and D. Shen, "State-space model with deep learning for functional dynamics estimation in resting-state FMRI," *Neuroimage*, Vol. 129, pp. 292–307, 2016.

[50] Y. Guo, G. Wu, L.A. Commander, S. Szary, V. Jewells, W. Lin, and D. Shent, "Segmenting hippocampus from infant brains by sparse patch matching with deep-learned features," In *International Conference on Medical Image Computing and Computer-Assisted Intervention* (*MICCAI 2014),* Vol. 17(Pt 2), pp. 308-315, 2014.

[51] A. Mansoor, J. Cerrolaza, R. Idrees, E. Biggs, M. Alsharid, R. Avery, and M.G. Linguraru, "Deep learning guided partitioned shape model for anterior visual path-way segmentation," *IEEE Transactions on Medical Imaging*, Vol. 35, No. 8, pp. 856–1865, 2016.

[52] A. Benou, R. Veksler, A. Friedman, and T.R. Raviv, "De-noising of contrast-enhanced MRI sequences by an ensemble of expert deep neural networks," In: *Deep Learning and Data Labeling for Medical Applications;* Springer, Cham, pp. 95-110, 2016.

[53] J. Xu, L. Xiang, Q. Liu, H. Gilmore, J. Wu, J. Tang, and A. Madabhushi, "Stacked sparse autoencoder (SSAE) for nuclei detection on breast cancer histopathology images," *IEEE Transactions on Medical Imaging,* Vol. 35, No. 1, pp. 119–130, 2015.

[54] A. Janowczyk, A. Basavanhally, and A. Madabhushi, "Stain normalization using sparse autoencoders (StaNoSA): application to digital pathology," *Computerized Medical Imaging and Graphics,* Vol. 57, pp. 50–61, 2017.

[55] M. Kallenberg, K. Petersen, M. Nielsen, A. Ng, P. Diao, C. Igel, et al., “Unsupervised deep learning applied to breast density segmentation and mammographic risk scoring,” *IEEE Transactions on Medical Imaging*, Vol. 35, No. 5, pp. 1322–1331, 2016.

[56] Y. Zhu, L. Wang, M. Liu, C. Qian, A. Yousuf, A. Oto, and D. Shen, “MRI Based prostate cancer detection with high-level representation and hierarchical classification,” *Medical Physics,* Vol. 44, No. 3, pp. 1028–1039, 2017.

[57] N. Hatipoglu, and G. Bilgin, “Cell segmentation in histopathological images with deep learning algorithms by utilizing spatial relationships,” *Medical & Biological Engineering & Computing*, Vol. 55, No. 10, pp. 1829–1848, 2017.

[58] M.R. Avendi, A. Kheradvar, and H. Jafarkhani, “Automatic segmentation of the right ventricle from cardiac MRI using a learning-based approach,” *Magnetic Resonance in Medicine*, Vol. 78, No. 6, pp. 2439–2448, 2017.

[59] H. Su, F. Xing, X. Kong, Y. Xie, S. Zhang, and L. Yang, “Robust cell detection and segmentation in histopathological images using sparse reconstruction and stacked denoising autoencoders,” *Lecture Notes in Computer Science*, Vol. 9351, Springer, Cham, 2015.

[60] S. Liu, S. Liu, W. Cai, S. Pujol, R. Kikinis, and D. Feng, “Early diagnosis of Alzheimer’s disease with deep learning,” In *Proceedings of the IEEE 11th International Symposium on Biomedical Imaging (ISBI)*, pp. 1015–1018, 2014.

[61] J.Z. Cheng, D. Ni, Y.H. Chou, et al., “Computer-aided diagnosis with deep learning architecture: applications to breast lesions in US images and pulmonary nodules in CT scans,” *Scientific Reports*, Vol. 6, 24454, 2016.

[62] R. Miotto, L. Li, B.A. Kidd, and J.T. Dudley, “Deep patient: an unsupervised representation to predict the future of patients from the electronic health records,” *Scientific Reports*, Vol. 6, 26094, 2016.

[63] X. Cheng, L. Zhang, and Y. Zheng, “Deep similarity learning for multimodal medical images,” *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization*, Vol. 6, No. 3, pp. 248-252, 2018.

[64] H. Huang, X. Hu, Y. Zhao, M. Makkie, Q. Dong, S. Zhao, and T. Liu, “Modeling task fMRI data via deep convolutional autoencoder,” *IEEE Transactions on Medical Imaging*, Vol. 37, No. 7, pp. 1551-1561, 2018.

[65] E. Hosseini-Asl, G. Gimel’farb, and A. El-Baz, “Alzheimer’s disease diagnostics by a deeply supervised adaptable 3D convolutional network,” *arXiv preprint,* arxiv:1607.00556, 2016.

[66] L. Hou, V. Nguyen, A.B. Kanevsky, D. Samaras, T.M. Kurc, T. Zhao, et al., “Sparse autoencoder for unsupervised nucleus detection and representation in histopathology images,” *Pattern Recognition*, Vol. 86, pp. 188-200, 2019.

[67] P. Foggia, G. Percannella, A. Saggese, and M. Vento, “Pattern recognition in stained HEp-2 cells: Where are we now?,” *Pattern Recognition*, Vol. 47, pp. 2305–2314, 2014.

[68] S.D. Cataldo, A. Bottino, E. Ficarra, and E. Macii, “Applying textural features to the classification of HEp-2 cell patterns in IIF images,” In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR 2012)*, pp. 689–694, 2012.

[69] A. Wiliem, Y. Wong, C. Sanderson, P. Hobson, S. Chen, and B.C. Lovell, “Classification of human epithelial type 2 cell indirect immunofluorescence images via codebook based descriptors,” In *Proceedings of the IEEE Workshop on Applications of Computer Vision (WACV 2013)*, pp. 95–102, 2013.

[70] R. Stoklasa, T. Majtner, and D. Svoboda, “Efficient k-NN based HEp-2 cells classifier,” *Pattern Recognition,* Vol. 47, pp. 249–2418, 2014.

[71] R. Nosaka and K. Fukui, “HEp-2 cell classification using rotation invariant co-occurrence among local binary patterns,” *Pattern Recognition,* Vol. 47, pp. 2428–2436, 2014.

[72] S.D. Cataldo, A. Bottino, I.U. Islam, T.F. Vieira, and E. Ficarra, “Subclass discriminant analysis of morphological and textural features for HEp-2 staining pattern classification,” *Pattern Recognition,* Vol. 47, pp. 2389-2399, 2014.

[73] I. Theodorakopoulos, D. Kastaniotis, G. Economou, and S. Fotopoulos, "HEp-2cells classification via sparse representation of textural features fused into dissimilarity space," *Pattern Recognition,* Vol. 47, pp. 2367–2378, 2014.

[74] Y.C. Huang, T.Y. Hsieh, C.Y. Chang, W.T. Cheng, Y.C. Lin, and Y.L. Huang, "HEp-2 cell images classification based on textural and statistic features using self-organizing map," In *Proceedings of the 4th Asian Conference on Intelligent Information and Database Systems, Part II*, pp. 529–538, 2012.

[75] G. Thibault, J. Angulo, and F. Meyer, "Advanced statistical matrices for texture characterization: Application to cell classification," *IEEE Transactions on Biomedical Engineering,* Vol. 61, pp. 630–637, 2013.

[76] A. Wiliem, C. Sanderson, Y. Wong, P. Hobson, R.F. Minchin, and B.C. Lovell, "Automatic classification of human epithelial type 2 cell indirect immunofluorescence images using cell pyramid matching," *Pattern Recognition,* Vol. 47, pp. 2315–2324, 2014.

[77] X. Xu, F. Lin, C. Ng, and K.P. Leong, "Automated classification for HEp-2 cells based on linear local distance coding framework," *EURASIP Journal on Image and Video Processing,* Vol. 2015, pp. 1–13, 2015.

[78] G.V. Ponomarev, V.L. Arlazarov, M.S. Gelfand, and M.D. Kazanov, "ANA HEp-2 cells image classification using number, size, shape and localization of targeted cell regions," *Pattern Recognition,* Vol. 47, pp. 2360–2366, 2014.

[79] L. Shen, J. Lin, S. Wu, and S. Yu, "HEp-2 image classification using intensity order pooling based features and bag of words," *Pattern Recognition,* Vol. 47, pp. 2419–2427, 2014.

[80] Z. Gao, L. Wang, L. Zhou, and J. Zhang, "HEp-2 cell image classification with deep convolutional neural networks," *IEEE Journal of Biomedical and Health Informatics*, Vol. 21, pp. 416–428, 2017.

[81] N. Bayramoglu, J. Kannala, and J. Heikkilä, "Human epithelial type 2 cell classification with convolutional neural networks," In *Proceedings of the IEEE*

15th International Conference on Bioinformatics and Bioengineering (BIBE 2015), pp. 1–6, 2015.

[82] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," In *Proceedings of the 22nd ACM International Conference on Multimedia*, pp. 675–678, 2014.

[83] J. Liu, B. Xu, L. Shen, J. Garibaldi, and G. Qiu, "HEp-2 cell classification based on a deep autoencoding-classification convolutional neural network," In *Proceedings of the 14th IEEE International Symposium on Biomedical Imaging (ISBI 2017)*, pp. 1019–1023, 2017.

[84] L.F. Rodrigues, M.C. Naldi, and J.F. Mari, "Comparing convolutional neural networks and preprocessing techniques for HEp-2 cell classification in immunofluorescence images," *Computers in Biology and Medicine*, Vol. 116, 103542, 2020.

[85] J. Xi, S. Linlin, Z. Xiande, and Y. Shiqi, "Deep convolutional neural network based HEp-2 cell classification," In *Proceedings of the 23rd International Conference on Pattern Recognition (ICPR 2016)*, pp. 77–80, 2016.

[86] Y. Li and L. Shen, "A deep residual inception network for HEp-2 cell classification," In *Proceedings of 3rd International Workshop, DLMIA 2017* and *7th International Workshop, ML-CDS 2017*, 2017.

[87] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016)*, pp. 770–778, 2016.

[88] C. Szegedy, W. Liu, Y. Jia, and P. Sermanet, "Going deeper with convolutions," In *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2015)*, pp. 1–9, 2015.

[89] L. Shen, X. Jia, and Y. Li, "Deep cross residual network for HEp-2 cell staining pattern classification," *Pattern Recognition,* Vol. 82, pp. 68–78, 2018.

[90] T. Majtner, B. Bajic, J. Lindblad, N. Sladoje, V. Blanes-Vidal, and E.S. Nadimi, "On the effectiveness of generative adversarial networks as HEp-2 image augmentation tool," In *Proceedings of the Scandinavian Conference on Image Analysis (SCIA 2019)*, pp. 439–451, 2019.

[91] I.J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," In *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS 2014)*, pp. 2672–2680, 2014.

[92] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," In *Proceedings of the 4th International Conference on Learning Representations (ICLR 2016)*, 2016. Available online: https://arxiv.org/pdf/1511.06434.pdf, Accessed on July 21, 2021.

[93] Y. Li and L. Shen, "HEp-Net: A smaller and better deep-learning network for HEp-2 cell classification," *Computer Methods in Biomechanics and Biomedical Engineering,* Vol. 7, pp. 266–272, 2018.

[94] H.T.H Phan, A. Kumar, J. Kim, and D. Feng, "Transfer learning of a convolutional neural network for HEp-2 cell image classification," In *Proceedings of the 2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI 2016)*, pp. 1208–1211, 2016.

[95] K. Simonyan, and A. Zisserman, "A very deep convolutional networks for large-scale image recognition," In *Proceedings of the 2015 International Conference on Learning Representation (ICLR15)*, 2015. Available online: https://arxiv.org/pdf/1409.1556. pdf, Accessed on July 21, 2021.

[96] M. Lu, L. Gao, X. Guo, Q. Liu, and J. Yin, "HEp-2 cell image classification method based on very deep convolutional networks with small datasets," In *9th International Conference on Digital Image Processing (ICDIP 2017)*, 2017.

[97] L.D. Nguyen, R. Gao, D. Lin, and Z. Lin, “Biomedical image classification based on a feature concatenation and ensemble of deep CNNs,” *Journal of Ambient Intelligence and Humanized Computing,* Vol. 1, 2019.

[98] D. Cascio, V. Taormina, and G. Raso, “Deep CNN for IIF images classification in autoimmune diagnostics,” *Applied Sciences,* Vol. 9, 1618, 2019.

[99] A. Krizhevsky, I. Sutskever, and G.E. Hinton, “ImageNet classification with deep convolutional neural networks,” In *Proceedings of the 25th International Conference on Neural Information Processing Systems (NIPS'12)*, pp. 1097–1105, 2012.

[100] H. Lei, T. Han, F. Zhou, Z. Yu, J. Qin, A. Elazab, and B. Lei, “A deeply supervised residual network for HEp-2 cell classification via cross-modal transfer learning,” *Pattern Recognition,* Vol. 79, pp. 290–302, 2018.

[101] B.C. Lovell, G. Percannella, A. Saggese, M. Vento, and A. Wiliem, “International contest on pattern recognition techniques for indirect immunofluorescence images analysis,” In *Proceedings of the 2016 23rd International Conference on Pattern Recognition (ICPR 2016)*, pp. 74–76, 2016.

[102] B. Yang, X. Fu, N.D. Sidiropoulos, and M. Hong, “Towards k-means-friendly spaces: Simultaneous deep learning and clustering,” In *Proceedings of the 34th International Conference on Machine Learning (ICML 2017)*, pp. 3861–3870, 2017. Available online: https://arxiv.org/pdf/1610.04794.pdf, Accessed on July 21, 2021.

[103] V. Badrinarayana, A. Kendall, and R. Cipolla, “SegNet: A deep convolutional encoder-decoder architecture for Image Segmentation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 39, pp. 2481–2495, 2017.

[104] O. Ronneberger, P. Fischer, and T. Brox, “U-Net: convolutional networks for biomedical image segmentation,” In *Proceedings of the 18th International Conference on Medical Image Computing and Computer-Assisted Intervention—MICAAI 2015*, pp. 234–241, 2015.

[105] W. Wang and J. Shen, “Deep visual attention prediction,” *IEEE Transactions on Image Processing*, Vol. 27, No. 5, pp. 2368-2378, 2018.

[106] D. Cascio, V. Taormina, and G. Raso, “Deep convolutional neural network for HEp-2 fluorescence intensity classification,” *Applied Sciences,* Vol. 9, 408, 2019.

[107] M. Merone, C. Sansone, and P. Soda, “A computer-aided diagnosis system for HEp-2 fluorescence intensity classification,” *Artificial Intelligence in Medicine*, Vol. 97, pp. 71–78, 2019.

[108] I. Nigam, S. Agrawal, R. Singh, and M. Vatsa, “Revisiting HEp-2 cell classification,” *IEEE Access,* Vol. 3, pp. 3102–3113, 2015.

[109] C. Vununu, S.K. Lee, O.J. Kwon, and K.R. Kwon, “A dynamic learning method for the classification of the HEp-2 cell images,” *Electronics,* Vol. 8, 850, 2019.

[110] D.E. Rumelhart, G.E. Hinton, and R.J. Williams, “Learning representations by back-propagating errors,” *Nature*, Vol. 323, pp. 533–536, 1986.

[111] K. He, X. Zhang, S. Ren, and J. Sun, “Identity mappings in deep residual networks,” In *Proceedings of the 14th European Conference on Computer Vision (ECCV 2016)*, pp. 630–645, 2016.

[112] D. Cai, X. He, and J. Han, “Locally consistent concept factorization for document clustering,” *IEEE Transactions on Knowledge and Data Engineering,* Vol. 23, pp. 902–913, 2011.

[113] K.Y. Yeung and W.L. Ruzzo, “Details of the adjusted rand index and clustering algorithms, supplement to the paper an empirical study on principal component analysis for clustering gene expression data,” *Bioinformatics,* Vol. 17, pp. 763–774, 2001.

[114] X. Qi, G. Zhao, J. Chen, and M. Pietikainen, “Exploring illumination robust descriptors for human epithelial type 2 cell classification,” *Pattern Recognition*, Vol. 60, pp. 420-429, 2016.

[115] S. Sinha, S. Ebrahimi, and T. Darrell, "Variational adversarial active learning," In *Proceedings of the International Conference on Computer Vision (ICCV 2019)*, pp. 5972-5981, 2019.

[116] C. Vununu, S.K. Lee and K.R. Kwon, "A strictly unsupervised deep learning method for HEp-2 cell image classification," *Sensors*, Vol. 20, 2717, 2020.

[117] C. Vununu, S.K. Lee and K.R. Kwon, "A classification method for the cellular images based on active learning and cross-modal transfer learning," *Sensors*, Vol. 21, 1469, 2021.

[118] C. Vununu, K.W. Kang, S.K. Lee, and K.R. Kwon, "Pyramidal deep neural networks for the accurate segmentation and counting of cells in microscopy data," *Journal of Korea Multimedia Society*, Vol. 22, No. 1, pp. 101-114, 2018.

[119] C. Vununu, O.H. Kwon, K.S. Moon, C.Y. Kim, S.K. Lee, and K.R. Kwon, "Segmentation of mitochondria in electron microscopy using deep learning based feature generation and support vector machine," In *Proceedings of the 2018 International Symposium on Nonlinear Theory and its Applications (NOLTA 2018)*, pp. 621-624, 2018.

[120] C. Vununu, O.H. Kwon, K.W. Kang, S.K. Lee, and K.R. Kwon, "A deep feature learning scheme for counting the cells in microscopy data," In *Proceedings of the 2018 IEEE International Conference on Electronics and Communication Engineering (ICECE 2018)*, pp. 22-26, 2018.

[121] C. Vununu, K.S. Moon, S.K. Lee, K.W. Kang, and K.R. Kwon, "An unsupervised feature learning method for the classification of the cellular images," *IW-FCV 2019*, Seoul, Korea, January 2019.

[122] C. Vununu, S.K. Lee, and K.R. Kwon, "A deep-learning based method for the classification of the cellular images," in *Proceedings of the 13th International Conference on Biomedical Engineering Systems and Technologies (BIOSTEC 2020)*, pp. 242-245.

[123] C. Vununu, K.S. Moon, S.K. Lee, S.K. Kwon, and K.R. Kwon, “A classification method for the cell images based on convolutional autoencoders and artificial neural networks,” *IEEK 2020*, Jeju, Korea, August 2020.

[124] C. Vununu, K.S. Moon, S.K. Kwon, S.K. Lee, and K.R. Kwon, “Automated segmentation of mitochondria based on multiscale deep networks,” *KMMS Fall Conference 2020*, Seoul, Korea, November 2020.

VII. ACKNOWLEDGEMENT

Firstly, I would like to send my gratitude to my parents and all the members of my family for their constant encouragements during all these years I have been in the Republic of Korea.

Secondly, I would like to send my sincere thanks to my principal advisor, Prof. Ki-Ryong Kwon, for his constant guidance as well as for providing necessary means, information and direction for the accomplishment of the present work. I would like to express all my sincere acknowledgements for everything he has done for me. For all the encouragements, motivation and supports he has provided me, I will be forever thankful.

Special thanks to Prof. Suk-Hwan Lee for his great and effective help during every step of my time in the MCSP lab and during the writing process of the present work. I will forever remember all the interesting discussions and recommendations that made this work possible.

Also, to all the professors that have participated in the detailed review process of this thesis, Prof. Bong-Kee Sin, Prof. Oh-Heum Kwon, and Prof. Heung-Kook Choi, I present my sincere gratitude. Thanks also to all the professors that I have had during my courses in PKNU.

I take the opportunity to send my deeply warm gratitude to all the lab members that have been with me during my time in the MCSP Lab, from Sukhrob Atoev, Rafiul Hasan and Song Mingyu (송민규), my fellow friends, to 박진혁 and 이경민 박사, my seniors. Also, special thanks to Hoang Long, Farkhodov Khurshedjon, and 임선자 박사.