Journal of Biomolecular Screening
2016, Vol. 21(3) 252–259
© 2016 Society for Laboratory
Automation and Screening
DOI: 10.1177/1087057115623451
jbx.sagepub.com
SAGE

# High-Content Analysis of Breast Cancer Using Single-Cell Deep Transfer Learning

**Chetak Kandaswamy[1–3], Luís M. Silva[1,2,4], Luís A. Alexandre[5], and Jorge M. Santos[1,2,6]**

**Abstract**

High-content analysis has revolutionized cancer drug discovery by identifying substances that alter the phenotype of a cell, which prevents tumor growth and metastasis. The high-resolution biofluorescence images from assays allow precise quantitative measures enabling the distinction of small molecules of a host cell from a tumor. In this work, we are particularly interested in the application of deep neural networks (DNNs), a cutting-edge machine learning method, to the classification of compounds in chemical mechanisms of action (MOAs). Compound classification has been performed using image-based profiling methods sometimes combined with feature reduction methods such as principal component analysis or factor analysis. In this article, we map the input features of each cell to a particular MOA class without using any treatment-level profiles or feature reduction methods. To the best of our knowledge, this is the first application of DNN in this domain, leveraging single-cell information. Furthermore, we use deep transfer learning (DTL) to alleviate the intensive and computational demanding effort of searching the huge parameter's space of a DNN. Results show that using this approach, we obtain a 30% speedup and a 2% accuracy improvement.

**Keywords**

high-content screening, image analysis, deep transfer learning, cancer drug discovery

## Introduction

Recent advances in quantitative microscopy and high-performance computing have enabled rapid progress in the development of high-throughput image-based assays. These high-content analysis (HCA) assays allow not only a precise quantitative observation of multiple parameters such as nuclear size, nuclear morphology, DNA replication, and many more subtle features derived from each image, but also the screening of thousands of cells. To tackle this high-throughput high-dimensional problem, biologists tend to use population averages of per-cell information prior to machine learning (ML) algorithms such as principal component analysis, random forest, K-nearest neighbors, or support vector machines. Moreover, a recent survey[1] shows that about 70% of the papers on HCA experiments published in *Science, Nature, Cell*, and the *Proceedings of the National Academy of Sciences* from 2000 to 2012 used only one or two of the cell's measured features, and less than 15% used more than six. Unfortunately, and due to the exponential increase in the number of product terms,[2] such ML algorithms become impractical for these problems with thousands of samples and hundreds of measured features. As a result, about 85% of the research work in HCA underutilized potentially valuable information that might have helped in speeding up early-stage drug discovery. In this paper, we are interested in exploring state-of-the-art algorithms developed in the field of artificial intelligence to address these high-throughput high-dimensional data.

The discovery of hierarchical visual sensory processing systems in the neocortex of the mammal brain motivated the field of artificial intelligence to develop algorithms to hierarchically extract information from data.[3,4] Deep learning[5,6] has

[1]Instituto de Engenharia Biomédica (INEB), Porto, Portugal
[2]Instituto de Investigação e Inovação em Saúde, Universidade do Porto, Porto, Portugal
[3]Departamento de Engenharia Eletrotécnica e de Computadores, Faculdade de Engenharia da Universidade do Porto, Porto, Portugal
[4]Departamento de Matemática, Universidade de Aveiro, Aveiro, Portugal
[5]Universidade da Beira Interior, Instituto de Telecomunicações, Covilhã, Portugal
[6]Departamento de Matemática, Instituto Superior de Engenharia do Instituto Politécnico do Porto, Porto, Portugal

Received Mar 19, 2015, and in revised form Nov 26, 2015. Accepted for publication Nov 30, 2015.

Supplementary material for this article is available on the *Journal of Biomolecular Screening* Web site at http://jbx.sagepub.com/supplemental.

**Corresponding Author:**
Chetak Kandaswamy, Instituto de Engenharia Biomédica (INEB), Rua do Campo Alegre, 823, Porto, 4150-180, Portugal.
Email: chetak.kand@gmail.com

thus emerged as a new paradigm in artificial intelligence focusing on computational models for information representation that exhibit characteristics similar to those of the neocortex, in an attempt to imitate a primate visual system with its sequence of processing stages: detection of edges, primitive shapes, and moving up gradually to more complex visual shapes.[7,8] Since 2006, deep learning research has been successful not only in academia but also in companies such as Google (image retrieval) and Facebook (face recognition). With many application domains, including image recognition[9,10] and speech recognition,[11] deep learning has beaten other ML techniques at predicting the activity of potential drug molecules using quantitative structures[12] and predicting the effects of mutations in noncoding DNA on gene expression and disease.[13]

We focus on the challenge of using information content as high as possible, by considering per-cell information and all the available features, to build a classifier for the chemical mechanism of action (MOA). A mechanism of action usually refers to biochemical interaction through which the drug binds to form pharmacological effects. In here, MOA is specifically used to express a share of similar phenotypic outcomes among different compound treatments and not a strict modulation of a particular target or target class.[14] According to Ljosa et al.,[14] the mechanistic classes were selected to provide the data with a wide cross section of cellular morphological phenotypes. We propose a deep transfer learning (DTL) framework, combining the advantages of deep learning with the flexibility of transfer learning. Transfer learning consists of reusing the knowledge gained from a (source) problem to solve a new (target) problem. Ideally, DTL should improve the performance of the reused classifier in the target problem over the baseline, that is, over the classifier trained directly in the target problem.

Our contribution can thus be summarized as follows:

1. Use of per-cell information with all the extracted features from high-content images
2. Use of state-of-the-art deep learning models coupled with GPU computational power to analyze such high-throughput high-dimensional data
3. Use of transfer learning to improve the performance of the models (in terms of computational speed)

In this paper, we consider stacked autoassociators[15,16] (SAAs) as classifiers of MOAs on freely available MFC7 wild-type breast cancer data[14] using a DTL framework that includes a supervised layer-based feature transference approach.[16,17]

A possible use case of the work presented in this paper would be for a researcher to (1) solve a given classification problem of MOA or obtain the classifiers used to solve such a problem from the result of a previous work, (2) select what part of a previously developed classifier to transfer, and (3) solve the new problem by doing transfer learning of the learned classifiers for a new MOA task and benefit from a faster training (when compared to a random initialization) and an eventual improvement in classification accuracy. In the case of using deep neural networks as classifiers, as we do in this work, the researcher can choose which layers should be reused from a previous experiment. In the Results section, we discuss several settings and advise the use of the setting that produces the best results in our work.

## Materials and Methods

### Data

We used a publicly available (http://www.broadinstitute.org/bbbc, accession BBBC021) dataset from the genetically engineered MCF7-wt (breast cancer expressing wild-type p53) cell line.[26] Briefly (all details of sample preparation and image analysis can be found in Ljosa et al.[14]), images of cell cultures with a given treatment (specific compound × concentration combination) were acquired on a high-content imaging platform using a 16-bit camera. Each image was further segmented using CellProfiler[18] (CP) by identifying nuclear and cytoplasmic boundaries. Then, 453 distinct features for each cell representing a variety of geometric, intensity, subcellular localization, and texture features[19] were extracted with CP. **Figure 1** shows some examples of captured images representing some of the MOAs, as well as some of the features extracted with CP.

Our problem consists of predicting the MOA of a given treatment using per-cell information, in contrast to other established methodologies that use some profiling and/or feature reduction techniques (see Ljosa et al.[14] for a comparative study). Profiling in this context is meant as the process of building a multivariate vector profile for each treatment based on all the cells treated with that treatment. There are a total of 103 treatments corresponding to combinations of 38 compounds at one to seven concentrations. We only used the 148,649 cells of noncontrol samples, thus giving a data matrix with 148,649 rows (representing cells) and 453 columns (representing the extracted features).

To perform transfer learning, we need to define a source and a target problem. For that purpose, the original MFC7 dataset with 12 MOAs is split into two mutually exclusive datasets with 6 MOAs each, $P_{set1}$ and $P_{set2}$. The distribution across the two subsets was performed in order to join MOAs with common batches (a batch represents the week in which a group of cells were cultured in the same environmental setting) to prevent classification bias arising from batch and/or plate effects (see **Suppl. Table S1** for more details).

### Classifier: Stacked Autoassociators

Let us represent a dataset by a set of tuples $D = \{(\boldsymbol{x}_i, y_i) \in X \times Y\}$, $i = 1, \ldots, n$, where $X$ is the input

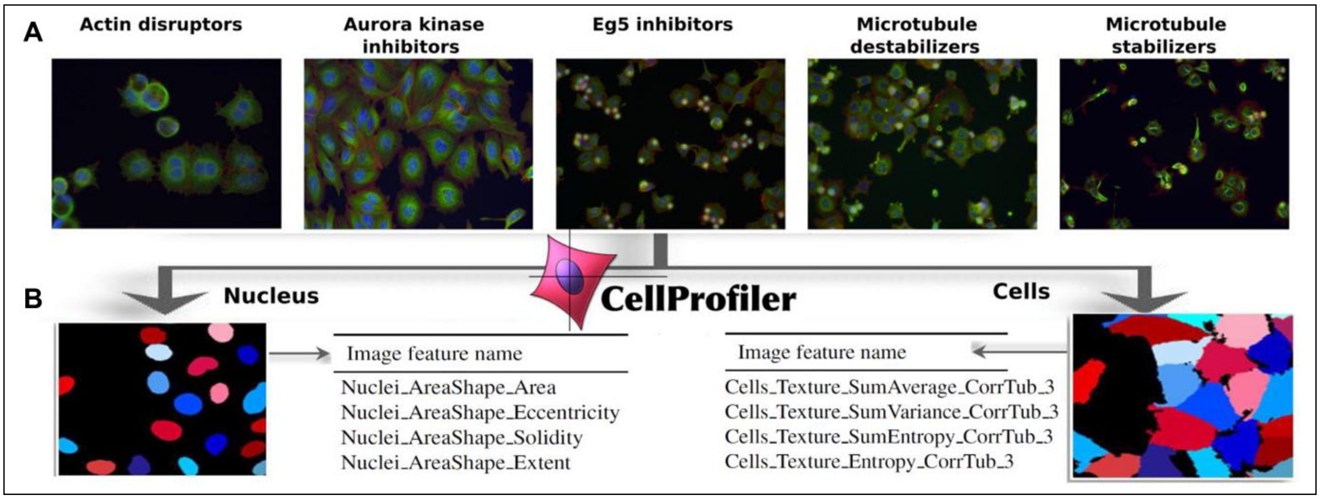
A
Actin disruptors
Aurora kinase inhibitors
Eg5 inhibitors
Microtubule destabilizers
Microtubule stabilizers
B
Nucleus
CellProfiler
Cells
Image feature name
Nuclei_AreaShape_Area
Nuclei_AreaShape_Eccentricity
Nuclei_AreaShape_Solidity
Nuclei_AreaShape_Extent
Image feature name
Cells_Texture_SumAverage_CorrTub_3
Cells_Texture_SumVariance_CorrTub_3
Cells_Texture_SumEntropy_CorrTub_3
Cells_Texture_Entropy_CorrTub_3

**Figure 1.** (**A**) Examples of different phenotypes (MOAs) captured after compound incubation of MFC7-wt cells. According to Ljosa et al.,[14] only 6 of the 12 MOAs were visually identifiable. (**B**) Cell segmentation and feature extraction are performed using CellProfiler.[18] For each cell, a variety of geometric, intensity, subcellular localization, and texture features were extracted.

space and $Y$ is a set of label codes. Assume that the $n$ instances of the dataset are drawn by a sampling process from the input space $X$ with a certain probability distribution $P(X)$. A classifier is any function $g(\boldsymbol{x}): X \rightarrow Y$ that maps instances $\boldsymbol{x} \in X$ to label codes in $Y$. The correspondence between the label set $\Omega$ and the coding set $Y$ is defined by some one-to-one mapping (e.g., $\Omega = \{Actin\ disruptors, ..., Protein\ synthesis\} \rightarrow Y = \{1, ..., 12\}$, where $|\Omega| = 12$ is the cardinality of $\Omega$).

In this paper, we consider stacked autoassociators[24] (SAA) to build our classifier of MOAs. An autoencoder or autoassociator is a simple neural network with one hidden layer designed to reconstruct its own input. We additionally constrain the encoding and decoding feature sets (input-hidden and hidden-output weights, respectively) to be the transpose of each other (tied weights). SAA training[17] comprises two stages: an unsupervised *pretraining* stage where the information of the labels (MOAs) is not used, followed by a supervised *fine-tuning* stage, now using the MOA information. In the pretraining stage, a greedy layerwise approach is used to train the hidden layers of the SAA. The first hidden layer $\boldsymbol{h}_1$ is considered a regular autoassociator and its features (weights) $\{\boldsymbol{w}^1, (\boldsymbol{w}^1)^T\}$ are trained for several epochs in order to reconstruct the original inputs. After the first layer is pretrained, we keep only the encoding features $\boldsymbol{w}^1$ and stack a second (hidden) layer $\boldsymbol{h}_2$ over $\boldsymbol{h}_1$with weights $\{\boldsymbol{w}^2, (\boldsymbol{w}^2)^T\}$ that are trained in a similar way, but now to reconstruct the $\boldsymbol{h}_1$ values. This process is repeated until the $k$ th hidden layer is pretrained. In the fine-tuning stage, a logistic regression layer $\boldsymbol{h}_1$ with $|\ \Omega\ |$ neurons and weight vector $\boldsymbol{w}^1$ is added to the top of the pretrained machine, and this entire network is fine-tuned using the training subset (now with the labels) in order to minimize a cross-entropy loss function measuring the error between the classifier's predictions and the correct label codes. The optimization process uses a stochastic gradient descent approach of backpropagation using batches of training data to speed up computation time. The learned features are represented by the weights and biases of the trained SAA. For an SAA with $k$ hidden layers, $\boldsymbol{W} = \{\boldsymbol{w}^1, \boldsymbol{w}^2, \ldots, \boldsymbol{w}^k, \boldsymbol{w}^l\}$ is the set of all such parameters. **Figure 2** describes these two stages.

## Framework: Deep Transfer Learning

Traditionally, the goal of transfer learning is to transfer the knowledge (learning) obtained with a *source* problem to one or more *target* problems to efficiently develop an effective hypothesis for a new task, problem, or distribution.[20]

In this work, we combine deep learning with transfer learning by means of a supervised layer-based feature transference[21,22] method. In this method, a deep classifier is obtained (pretrained and fine-tuned) using data from a source problem and reused (partially or not) in the deep classifier for the target problem. The latter is finally fine-tuned with the data from the target problem. By partially, we mean that one can transfer all or part of the source model features (layers) to the target model. In this way, we are transferring knowledge acquired with the source to help in solving the target. It is expected that the TL process supplies the target classifier with an initial set of weights that is a better starting point than the traditional random initialization, providing improved performance (*positive transference*) over the baseline (by contrast, *negative transference* occurs when the baseline classifier performs better than the TL classifier). To be more precise, let us introduce some notation considering an SAA with seven hidden layers plus one logistic layer, for both the source and target models. We use four different TL settings for supervised layerwise

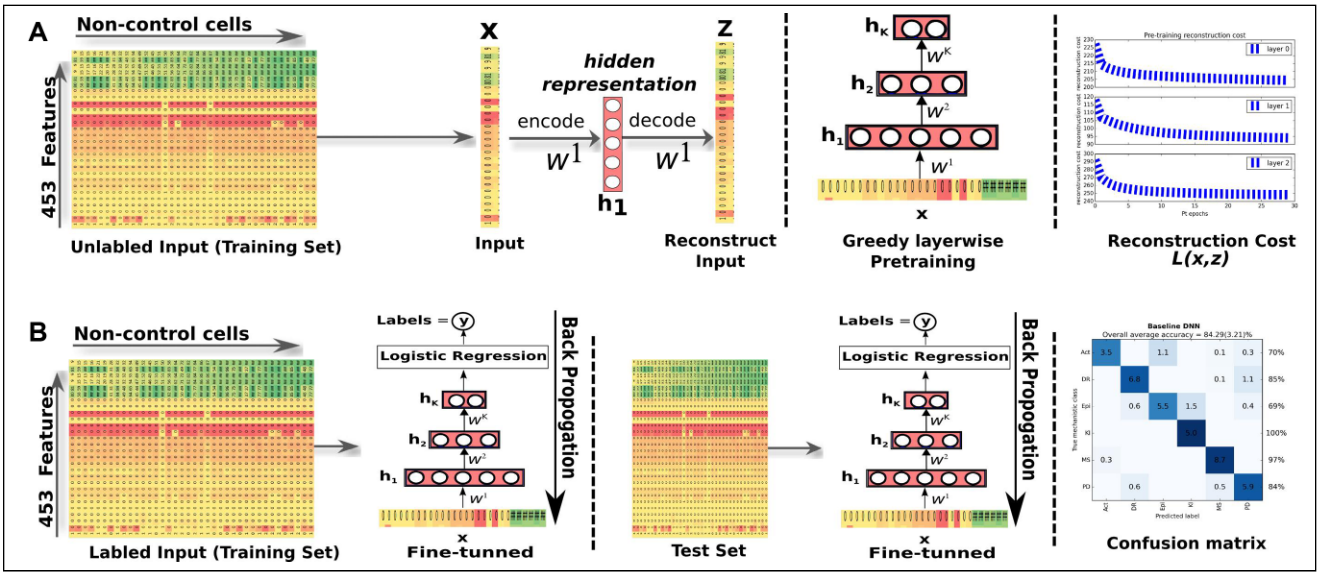

**Figure 2.** High-content image analysis of breast cancer cells using SAA. (**A**) Process of unsupervised greedy layerwise pretraining. The features of each cell are encoded into a hidden representation and reconstructed by minimizing reconstruction cost *L* (*x*, *z*). The hidden representation is then used as input for the next layer and the process repeated until the *k* th hidden layer is completely pretrained. (**B**) Process of supervised fine-tuning and baseline classifier performance evaluation on the test set.

feature transference. In such settings the 0 represents "no transfer," that is, the weights of that specific layer of the target model are randomly initialized and not reused from the source model, and the 1 represents "transferred," that is, the initial weights of that specific layer are obtained (reused) from the trained source model. Note that for each setting, the logistic regression layer is also transferred from the source model to the target model. The setting [00111111] means that we randomly initialized the first and second layers of the target model and transferred all the remaining layers from the source problem. The target network thus built is then fine-tuned with the target data.

## LOOCV Training and Network Hyperparameters

Regarding the training process, we followed a procedure similar to that in Ljosa et al.[14] To prevent sharing of batch-specific image properties/features or compound properties between the training and test sets, and thus to prevent the classifiers from learning artifact properties of the set of individual images rather than the more general cell phenotype,[23] we considered using a leave-one-compound-out cross-validation (LOOCV) procedure where all the cells treated with the same compound as the treatment being classified are held out, even if those other cells were treated with a different concentration. Thus, the test set in LOOCV is composed of all the cells from one of the compounds that is held out; the remaining cells (from all the other compounds) are split in a training set, used to train the model, and a validation set, used to prevent overfitting by evaluating *early-stopping criteria* in the fine-tuning phase. The choice of when to stop fine tuning is based on a geometrically increasing amount of patience. The patience is geometrically increased when the current validation score is below the best validation score. The backpropagation error is fine-tuned until it runs out of patience or the maximum fine-tuning epochs allowed is reached. The trained classifier is then tested on the unseen individual cells from the test set, and each prediction is matched with its ground truth of MOA. The classifier prediction of each cell from the same field of view is then combined to calculate treatment prediction accuracy using majority voting. Each of the experiments is repeated 10 times.

Tuning hyperparameters such as the learning rate or setting the appropriate network architecture for training the deep model is desirable but highly time-consuming. The results of the following section were obtained using SAAs with seven hidden layers of 500 neurons each. We used pretraining and fine-tuning learning rates of 0.001 and 0.1, respectively. The stopping criteria for pretraining were fixed to 60 epochs, which is the value where the reconstruction cost saturates; stopping criteria for fine tuning were set to a maximum of 1000 epochs with the validation set. The complete details of these networks are listed in **Supplementary Table S2**.

Processing large data as we did, on millions of neural connections, would take several weeks using traditional CPUs. For that reason, we used Theano,[24] a GPU-compatible machine learning library, to perform all our experiments on two i7-377 (3.50 GHz), 16 GB RAM with two GTX 770 and five GTX 980 GPU processors, respectively (see High-Performance Computing section of the supplementary material). The software to reproduce the results is available at http://www.deepnets.ineb.up.pt/files/software/DTL_frontend.html.

**Table 1.** Average Accuracy in Percentage and Average Computation Time in Minutes (Standard Deviation in Parentheses) of the Baseline (BL) and DTL Approaches.

| Settings | | | | | Test | | Time per Compound (min) | | |
|---|---|---|---|---|---|---|---|---|---|
| Approach | Transfer | $\boldsymbol{P_S}$ | $\boldsymbol{P_T}$ | $C$ | Accuracy | $p$ Value (to BL) | Pretrained | Fine-Tuned | Total Time per Repetition (min) |
| BL | | | $P_{set1}$ | 20 | 84.29 (3.21) | | 8.34 (0.0) | 16.98 (1.3) | 506 (29) |
| DTL_1 | [00000011] | $P_{set2}$ | $P_{set1}$ | 20 | **87.62 (6.96)** | 0.187 | — | 17.54 (2.5) | 350 (51) |
| DTL_2 | [00001111] | $P_{set2}$ | $P_{set1}$ | 20 | 77.62 (8.80) | 0.351 | | 15.08 (1.4) | 301 (29) |
| DTL_3 | [00111111] | $P_{set2}$ | $P_{set1}$ | 20 | 86.19 (8.73) | 0.589 | | 16.72 (2.0) | 334 (41) |
| DTL_4 | [11111111] | $P_{set2}$ | $P_{set1}$ | 20 | 86.43 (3.38) | 0.331 | | 10.35 (0.9) | 207 (18) |
| BL | | | $P_{set2}$ | 18 | 87.05 (4.25) | | 12.71 (0.2) | 26.10 (1.8) | 698 (37) |
| DTL_1 | [00000011] | $P_{set1}$ | $P_{set2}$ | 18 | **87.87 (6.86)** | 0.734 | | 27.36 (2.3) | 492 (42) |
| DTL_2 | [00001111] | $P_{set1}$ | $P_{set2}$ | 18 | 69.67 (11.4) | <0.001 | | 21.39 (2.7) | 385 (49) |
| DTL_3 | [00111111] | $P_{set1}$ | $P_{set2}$ | 18 | 85.08 (6.99) | 0.513 | | 25.33 (2.8) | 455 (50) |
| DTL_4 | [11111111] | $P_{set1}$ | $P_{set2}$ | 18 | 75.57 (4.72) | <0.001 | | 19.79 (2.2) | 356 (41) |

The results are over 10 repetitions for the target data ($P_T$) with compounds ($C$) and source data ($P_S$). The best results show in bold.

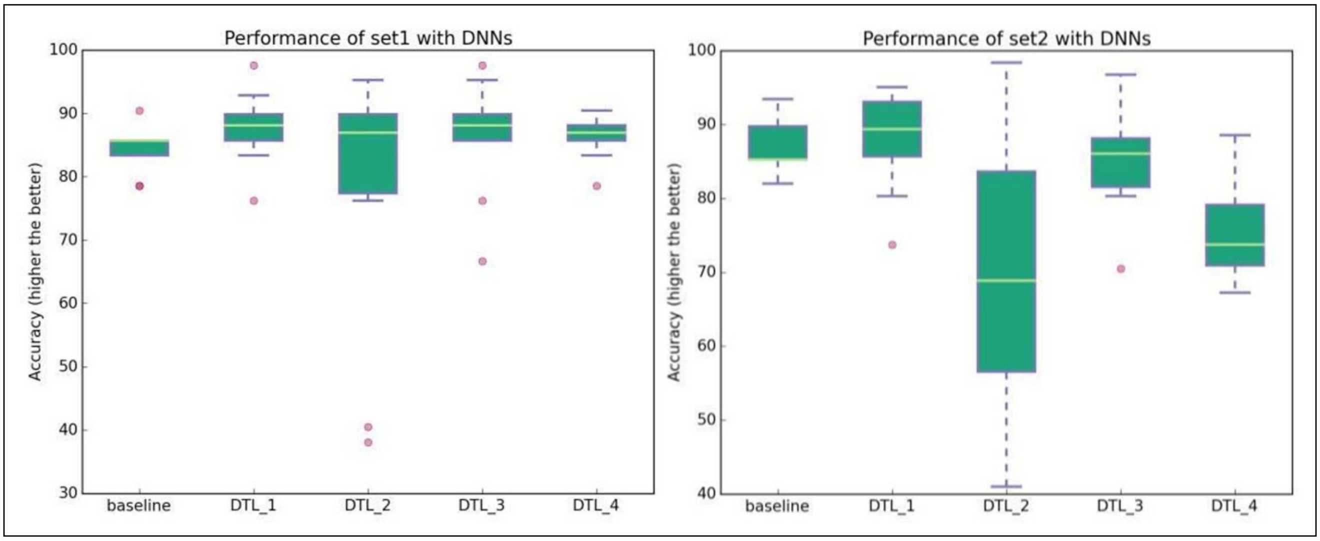

**Figure 3.** Comparison of baseline vs. DTL approaches. Left: Baseline average accuracy for classifying $P_{set1}$ and DTL approaches for classifying $P_{set1}$ reusing $P_{set2}$. Right: Baseline average accuracy for classifying $P_{set2}$ and DTL approaches for classifying $P_{set2}$ reusing $P_{set1}$.

## Results

The analysis of large volumes of multiparametric high-dimensional data without overfitting the network using a high number of cytological features in a time frame suitable for drug discovery presents a significant challenge for any learning algorithm. In the following, we present the results obtained by our approach.

The results of the baseline SAA for classifying MOAs for $P_{set1}$ and $P_{set2}$ datasets are listed in **Table 1**. We observe that classifying MOAs of $P_{set2}$ is about 2.8% more accurate than classifying MOAs of $P_{set1}$, even though both datasets have an equal number of MOAs. Also, the computation time to classify the $P_{set2}$ dataset is greater than that of the $P_{set1}$ dataset. The $P_{set2}$ dataset has 61 treatments for 18 compounds, whereas $P_{set1}$ has 42 treatments for 20 compounds. The confusion matrix for classifying MOAs using the baseline approach for both $P_{set1}$ and $P_{set2}$ datasets is shown in **Figure 4**, and the precision, recall, and $f1$-scores are listed in **Supplementary Table S3**.

To further improve the results over the baseline approach, we considered a deep transfer learning framework where the knowledge gained with the source problem is reused to solve the target problem. The results for four DTL settings are presented in **Table 1** and the respective boxplots displayed in **Figure 3**. Essentially, we observe that the DTL_1 setting improves over the baseline for both $P_{set1}$ and $P_{set2}$ datasets. It is interesting to note that the best results are obtained when such specific (top) layer weights are transferred from the source to the target problem (the seventh hidden layer weights and the logistic regression weights are reused) and the rest of the

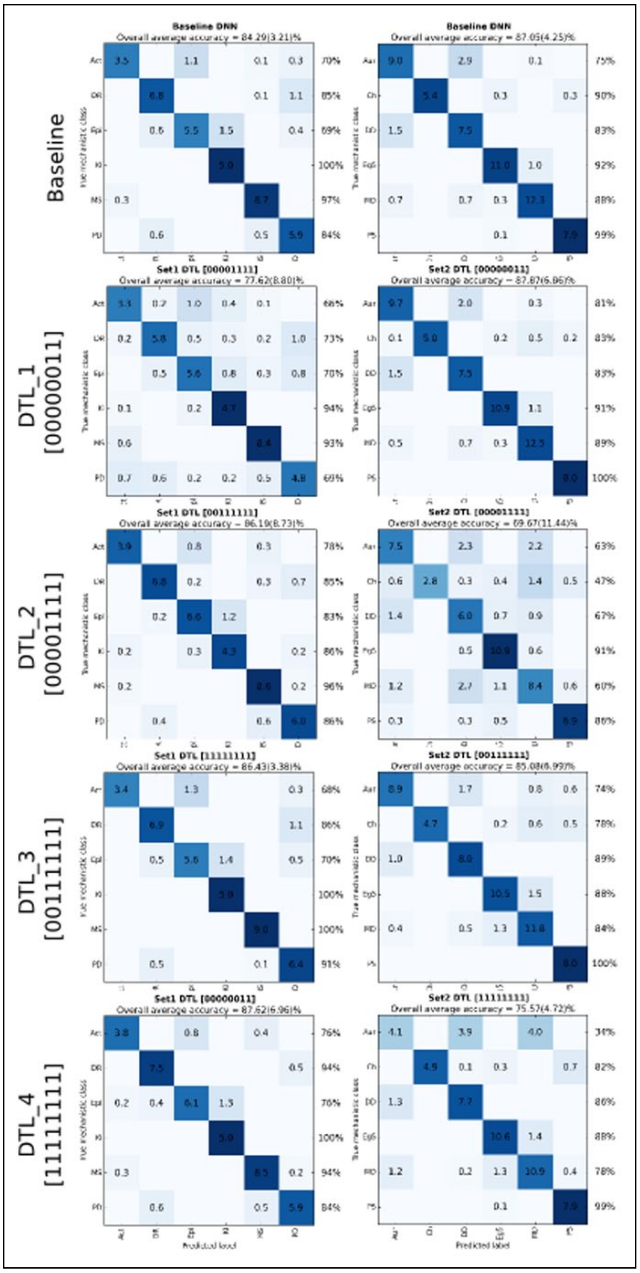

**Figure 4.** Confusion matrices for the baseline and TL settings on the MOA problem (average outcomes over 10 repetitions). To represent class imbalance, the confusion matrix represents number of elements in each class and the background blue color is normalized confusion matrices (the higher the accuracy, the darker the color).

(lower) layers are randomly initialized. For example, classifying $P_{set1}$ reusing $P_{set2}$ with the DTL_1 transfer setting produces models 2% more accurate than the baseline and about 0.8% over the *transfer all* case DTL_4. One of the reasons for this behavior is that higher layers of the network learn problem-specific features from the data, while the lower layers learn generic features;[21-22] thus, it seems beneficial to use the knowledge acquired in the source problem on its higher layers. Moreover, the DTL_1 setting speeds up computation time by 30% over the baseline approach. Confusion matrices for all DTL settings can be analyzed in **Figure 4**. Given these results, we believe that DTL_1 would be a good setting to use on similar problems by a researcher who wishes to use DTL on this type of problem.

### *Comparison with Other State-of-the-Art Methods*

**Table 2** lists a comparison of our deep learning (baseline and best TL setting) results with two state-of-the-art machine learning algorithms: support vector machines (SVMs)[25] with linear and radial basis function (RBF) kernels using a freely available and fast C-based implementation of multiclass SVM (SVM$^{multiclass}$, version 2.20). For linear SVM, we optimized the trade-off between training error and margin cost from 0.001 to 50,000 (see **Suppl. Table S4**), and the best model obtained an overall accuracy of about 21% for $P_{set1}$ and 23% for $P_{set2}$ (see **Suppl. Tables S7** and **S8**). For SVM RBF, we optimized the margin cost from 1 to 1000 and the gamma parameter from 0.001 to 0.00001 (see **Suppl. Table S5**). As the grid search is computationally expensive, we restricted to only one compound using 10% of the total training data. We observed the best model at margin cost 100 and gamma 0.001, but taking between 419 to 755 min to obtain a 45% accuracy (**see Suppl. Table S6**). Thus, we performed the experiments with 1% of the total training data to train the SVM RBF and obtained an overall accuracy of about 21% for $P_{set1}$ and 18% for $P_{set2}$ (see **Suppl. Tables S9** and **S10**). Further increasing the number of training samples improves the overall accuracy but leads to a high increase in computation time.

## Discussion

To stimulate the development of new drugs effective against a wide spectrum of cancers, we propose a deep transfer learning (DTL) classifying framework that uses high-content HCA data. Our classifiers are built upon individual cell information without employing any type of profiling or reduction methods on extracted cell features. The main motivation to use a DTL approach was to show that we can reuse, with minor modifications, the knowledge acquired in solving a given classification problem of MOAs to solve a new one (of MOAs also) *without having to follow the whole training procedure*. This is particularly useful for new drug testing, as computational time is saved. For that purpose, the data were carefully split into two mutually exclusive six-class problems represented by $P_{set1}$ and $P_{set2}$ datasets. The average accuracies of the baseline SAAs for the $P_{set1}$ and $P_{set2}$ datasets are about 84% and 87%, respectively, using a seven-hidden-layer SAA with 500 neurons in each layer. The DTL approach showed that the transference of

**Table 2.** Comparison of Accuracy Obtained and Total Time Taken per Repetition in Minutes with Other State-of-the-Art Methods.

| | $P_{set1}$ | | $P_{set2}$ | |
|---|---|---|---|---|
| Method | Accuracy (%) | Time (min) | Accuracy (%) | Time (min) |
| Linear SVM | 20.95 | 32 | 23.49 | 49 |
| SVM using RBF (model trained using 1% of total training data) | 21.04 | 78 | 17.50 | 125 |
| 8-layer-deep architecture (baseline) | 84.29 | 506 | 87.05 | 698 |
| DTL_1 [00000011] | **87.62** | 350 | **87.87** | 492 |

Best results in bold.

specific weights of the source model was useful, and we have obtained positive transference for both datasets. Although the difference in accuracy of $P_{set1}$ and $P_{set2}$ between baseline and transfer learning is not statistically significant, we observed around 30% computational speedup when using the DTL approach. Our approach was also superior when compared to multiclass support vector machines.

Regarding the 12-class problem, we trained several SAAs ranging from three to eight hidden layers with 500 to 1000 neurons in each layer. However, training a seven-hidden-layer SAA with 500 neurons in each layer may take, on average, 30–48 h per repetition. We performed some preliminary experiments using the adequate leave-one-out approach, and without too much hyperparameter search, the best model obtained around 77% accuracy. As future work, we intend to explore a different approach for the 12-class problem using convolutional neural networks (CNNs) directly applied to the images and not to hand-crafted features. CNNs are state-of-the-art deep neural networks that use a sort of hierarchical representation of the data similar to that of the neocortex and are especially designed for image recognition tasks. We expect to obtain a similar hierarchical feature extraction directly from the images, giving the possibility of the deep network self-extracting relevant cytological features layer by layer.

## Acknowledgments

We would like to acknowledge the support of Joaquim Marques de Sá, Jaime S. Cardoso, Vebjorn Ljosa, Shantanu Singh, Szymon Stoma, Tiago Laundos Santos, Jonathan Barber, Ricardo Sousa, and Abhishek Chatterjee. We also acknowledge the valuable comments provided by the reviewers that greatly helped to improve the paper.

## Declaration of Conflicting Interests

The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

## Funding

The authors disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was financed by FEDER funds through the Programa Operacional Factores de Competitividade—COMPETE and by Portuguese funds through FCT—Fundação para a Ciência e a Tecnologia in the framework of the project PTDC/EIA-EIA/119004/2010.

## References

1. Singh, S.; Carpenter, A. E.; Genovesio, A. Increasing the Content of High-Content Screening: An Overview. *J. Biomol. Screen*. **2014**, *19*, 640–650.
2. LeCun, Y.; Bottou, L.; Bengio, Y.; et al. Gradient-Based Learning Applied to Document Recognition. *Proc. IEEE* **1998**, *86*, 2278–2324.
3. Serre, T.; Wolf, L.; Poggio, T. In *Object Recognition with Features Inspired by Visual Cortex*, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, June 20–25, 2005; *IEEE: Piscataway, NJ*, **2005**; *Vol. 2*, pp 994–1000.
4. Lee, T. S.; Mumford, D.; Romero, R.; et al. The Role of the Primary Visual Cortex in Higher Level Vision. *Vision Res*. **1998**, *38*, 2429–2454.
5. LeCun, Y.; Bengio, Y.; Hinton, G. Deep Learning. *Nature* **2015**, *521*, 436–444.
6. Schmidhuber, J. Deep Learning in Neural Networks: An Overview. *Neural Netw*. **2015**, *61*, 85–117.
7. Lee, T. S.; Mumford, D. Hierarchical Bayesian Inference in the Visual Cortex. *J. Opt. Soc. Am. A* **2003**, *20*, 1434–1448.
8. Bengio, Y. Learning Deep Architectures for AI. *Found. Trends Mach. Learn*. **2009**, *2*, 1–127.
9. Krizhevsky, A.; Sutskever, I.; Hinton, G. E. In *ImageNet Classification with Deep Convolutional Neural Networks*, Advances in Neural Information Processing Systems 25—Proceedings of the 26th Conference on Neural Information Processing Systems, Lake Tahoe, NV, Dec 2012; Pereira, F., Burges, C. J. C., Bottou, L.; et al., Eds.; MIT Press: Cambridge, MA, **2012**.
10. Mnih, V.; Kavukcuoglu, K.; Silver, D.; et al. Human-Level Control through Deep Reinforcement Learning. *Nature* **2015**, *518*, 529–533.
11. Hinton, G.; Deng, L.; Yu, D.; et al. Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. *IEEE Signal Process. Mag*. **2012**, *29*, 82–97.
12. Ma, J.; Sheridan, R. P.; Liaw, A.; et al. Deep Neural Nets as a Method for Quantitative Structure–Activity Relationships. *J. Chem. Inf. Model*. **2015**, *55*, 263–274.

13. Leung, M. K. K.; Xiong, H. Y.; Lee, L. J.; et al. Deep Learning of the Tissue-Regulated Splicing Code. *Bioinformatics* **2014**, *30*, i121–i129.
14. Ljosa, V.; Caie, P. D.; ter Horst, R.; et al. Comparison of Methods for Image-Based Profiling of Cellular Morphological Responses to Small-Molecule Treatment. *J. Biomol. Screen*. **2013**, *18*, 1321–1329.
15. Vincent, P.; Larochelle, H.; Lajoie, I.; et al. Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. *J. Mach. Learn. Res*. **2010**, *11*, 3371–3408.
16. Amaral, T.; Silva, L. M.; Alexandre, L. A.; et al. In *Using Different Cost Functions to Train Stacked Auto-Encoders*, Proceedings of the 12th Mexican International Conference on Artificial Intelligence (MICAI), Mexico City, Mexico, Nov 24–30, **2013**; Castro, F., Gelbukh, A., González, M., Eds.; Springer: Berlin, **2013**.
17. Amaral, T.; Sá, J.; Silva, L.; et al. In *Improving Performance on Problems with Few Labelled Data by Reusing Stacked Auto-Encoders*, 11th International Conference Image Analysis and Recognition (ICIAR), Vilamoura, *Portugal*, Oct 22–24, 2014; Campilho, A., Kamel, M., Eds.; *Lecture Notes in Computer Science*; Springer: Berlin, **2014**; Part I, Vol. 8814.
18. Carpenter, A. E.; Jones, T. R.; Lamprecht, M. R.; et al. CellProfiler: Image Analysis Software for Identifying and Quantifying Cell Phenotypes. *Genome Biol*. **2006**, *7*, R100.
19. Young, D. W.; Bender, A.; Hoyt, J.; et al. Integrating High-Content Screening and Ligand-Target Prediction to Identify Mechanism of Action. *Nat. Chem. Biol*. **2008**, *4*, 59–68.
20. Ben-David, S.; Blitzer, J.; Crammer, K.; et al. A Theory of Learning from Different Domains. *Mach. Learn*. **2010**, *79*, 151–175.
21. Kandaswamy, C.; Silva, L. M.; Alexandre, L. A.; et al. In *Improving Transfer Learning Accuracy by Reusing Stacked Denoising Autoencoders*, Artificial Neural Networks and Machine Learning—24th International Conference on Artificial Neural Networks (ICANN), Hamburg, Germany, Sept 15–19, 2014; Wermter, S., Weber, C., Duch, W.; et al., Eds.; *Lecture Notes in Computer Science*; Springer: Berlin, **2014**; Vol. 8681.
22. Yosinski, J.; Clune, J.; Bengio, Y.; et al. In *How Transferable Are Features in Deep Neural Networks?* Advances in Neural Information Processing Systems 27—Proceedings of the 28th Conference on Neural Information Processing Systems, Montréal, Dec 8–13, 2014; Ghahramani, Z., Welling, M., Cortes, C., Eds.; MIT Press: Cambridge, MA, **2014**.
23. Shamir, L. Assessing the Efficacy of Low-Level Image Content Descriptors for Computer-Based Fluorescence Microscopy Image Analysis. *J. Microsc*. **2011**, *243*, 284–292.
24. Bergstra, J.; Breuleux, O.; Bastien, F.; et al. In *Theano: A CPU and GPU Math Expression Compiler*, Proceedings of the Python for Scientific Computing Conference (SciPy), Austin, TX, June 20–July 3, 2010; van der Walt, S., Millman, J., Ed.; **2010**; *Vol. 4*: Austin, TX.
25. Cortes, C.; Vapnik, V. Support-Vector Networks. *Mach. Learn*. **1995**, *20*, 273–297.
26. Caie, P. D.; Rebecca, E. W; Alexandra, I.; et al. High-Content Phenotypic Profiling of Drug Response Signatures across Distinct Cancer Cells. *Mol. Cancer Ther*. **2010**, *9*, 1913–1926.