Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: Day Month Year
Manuscript Category

OXFORD

Bioimage Informatics

Cellular structure image classification with small targeted training samples

Dali Wang [1,3,*], Zheng Lu [1,], Zi Wang [1], Yichi Xu [2], Anthony Santella [2], and Zhirong Bao [2,*]

[1]Department of Electrical Engineering and Computer Science, University of Tennessee, Knoxville, TN 37934, USA
[2]Developmental Biology Program, Sloan Kettering Institute, New York, NY 10065, USA
[3]Environmental Science Division, Oak Ridge National Laboratory, Oak Ridge, TN 37831, USA.

*To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: Different cell types often have distinct cell shapes. Collective shape changes usually indicate morphogenetic events and mechanisms. The identification and detection of collective cell shape changes in an extensive collection of 3D time-lapse images of complex tissues is an important step in assaying such mechanisms but is a tedious and time-consuming task. Machine learning provides new opportunities to automatically detect cell shape changes. However, it is time-consuming and sometimes challenging to generate sufficient training samples for pattern identification through deep learning.
Result: We present a deep learning approach with a minimal well-annotated training samples, and apply it to identify multicellular rosettes from 3D live images of the *C. elegans* embryo with fluorescently labeled cell membrane. Our strategy is to combine two popular approaches, namely feature transfer and generative adversarial networks (GAN), to boost image classification with small samples. Specifically, we use a GAN framework and conduct an unsupervised training in order to capture the general characteristics with 11250 unlabeled images. We then transfer the structure of the GAN discriminator into a new Alex-style neural network for further learning with several dozens of labeled samples. We conducted several experiments to show that with 10-15 well labeled rosette image and 30-40 randomly selected non-rosette images (that is around 20 % of the original training datasets) our approach can identified rosettes with over 80% accuracy, that is over 90% of the model accuracy achieved with the whole training dataset. We also establish a public benchmark dataset for rosette detection. This GAN-based transfer approach can be applied to study other cellular structures with minimal training samples.
Contact: dwang7@utk.edu, baoz@mskcc.org
Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Live microscopy and image processing are commonly used for cell dynamic investigation, cellular behavior quantification, and simulation-based hypothesis testing (Bao *et al.* (2006); Giurumescu *et al.* (2012); Kyoda *et al.* (2012); Wang *et al.* (2016, 2017, 2018)). The huge amount of microscope data generated during the studies presents unprecedented challenges for human-based, interactive data analysis. Advanced computing technology has been used in microscope data analysis (Jones *et al.* (2009)), however, the majority of these efforts require deep domain knowledge through a label-intensive annotation process. Nowadays, AI-based computer-vision provides a "model-free" approach to solving generic data problems, such as object identification. For example, convolutional neural networks (CNNs) are widely adopted for object classification and identification Krizhevsky *et al.* (2012); Szegedy *et al.* (2015); Simonyan and Zisserman (2014). Some well-known CNNs usually contain a large number of parameters, (e.g., more than 25 million in ResNet-50 network), that require large training datasets. Due to funding limitation and the scarcity of domain experts, it is a challenging to apply the traditional ways to (1) collect and establish comprehensive training

font is very small

datasets from 3D live images of *C. elegans* embryogenesis, and to (2) identify and detect a particular cellular structure.

[*** Biological significance of MCRs *****]**

We present a method for unique cellular structure image classification using 3D time-lapse datasets directly. Our learning process contains two steps: common cellular structure learning and target cellular structure learning. We adopt basic concepts within the generative adversarial networks (Goodfellow *et al.* (2014); Arjovsky and Bottou (2017); Odena *et al.* (2016); Li *et al.* (2018); Radford *et al.* (2015)) to speed up the common structure learning. Then we ingest a small quantity of target structure images into the learning system to improve the efficiency of target structure learning via transfer learning and regularization (Pan *et al.*, 2010; Noroozi and Favaro, 2016; Doersch *et al.*, 2015). To handle many technical issues associated with the learning procedures, we also worked on image noise removal and the quantification of the impact of neural network structures (depth, width, connection) and settings (sampling strategy, polling size, learning rate, regularization) on the microstructure classification and detection.

2 Methods

2.1 Dataset

2.1.1 Raw data

We use a set of *C. elegans* microscopy images that contains 45 embryos. The raw images are 512 x 512 size images which may contain one to three embryos. Raw images are arranged in sets, each set contains 300 image stacks which microscope take at 1-minute time interval shows the growth of embryos. Each stack is a pseudo 3D image that contains 30 slices showing a different level of the embryo. All the images are captured using the same microscopy setting.

2.1.2 Images sets for experiments

Since each raw image may contain more than one embryos, we first crop raw images into 128 x 128 images so that each of the 128x128 images contain complex global structure information of a single embryo. The 128x128 image also fits well into a single GPU of an NVIDIA computational platform (see more information at the end of this section). Technically, we write an ImageJ (Schneider *et al.* (2012)) macro for this task: For each embryo, we first mark its bounding box, then inside this bounding box, we randomly select 128x128 images. For each of these images, we apply a 3-D median filter and adjusting the brightness range to remove the image noise. Examples of a raw image and denoised image are shown in Fig. 1.

We select the raw data from a developmental period of 61-minute to 110-minute. The embryo structure is relative simple before 61-minute that a meaningful structural pattern of rosette is seldom observed. The embryo structure becomes too complicated after the 110-minute, the current denoising routine cannot generate images with sufficient quality for the network to learn. There is a recent work that has a good result on microscopy image denoising (Weigert *et al.* (2017)). However, the code, only available for CPU, takes a much longer time than our current denoising routine and was not used in our dataset prepossessing. For each image stack, we use images between slice 9 and slice 13 as these slices usually have the best imaging quality.

In the celegans embryogenesis, rosette is a special collective shape changes that usually indicate morphogenetic events and mechanisms, we would like to find an efficient way to identify and locate rosette in the 128x128 3D live microscopic images. Considering the average cell size during the above mentioned developmental period, the typical structure of muticelluar rosette, we identify 32x32 as the appropriate image size for classification and detection. We create two image datasets for deep learning experiments. The first dataset is a collection of unlabeled images for unsupervised image synthesis task. The second dataset is a manually labeled dataset for supervised image classification task for target cellular structures. For unlabeled dataset, we randomly sampled one image patch of size 32x32 at each slice of these image stack. So our unlabeled dataset contains 45 x 50 x 5 = 11250 image patches. The labeled dataset contains 78 manually-selected 32x32 rosette image from 45 different embryos. Each of the manually-annotated rosette images contains a multicelluar convergence center. We also randomly selected around 200 non-rosette images form the unlabelled datasets (see more information in section 3.2.1).

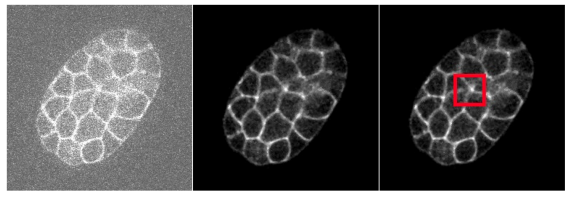


Fig. 1. Microscopy image of C. elegans before and after denoising. The red box in the left image represents the center of a rosette.

2.2 Neural network configuration

We modify an AlexNet-styled convolutional neural network (CNN) (Krizhevsky *et al.* (2012)) to classify the image (illustrated in 2). CNNs use convolution filters to automatically capture features rather than using hand-engineered features in traditional machine learning algorithms. The network has several convolutional layers (depends on the size of the input image), followed by two fully connected layers. When the input of our network is 32x32 grayscale image, our network has three convolutional layers. We use 4 x 4 filters for all convolutional layers. The number of filters at the first convolution layer is 32 and doubled at each convolutional layers. Unlike AlexNet, we replace all pooling layers with stride (2 pixels) convolutions so that the network can learn its own pooling method Springenberg *et al.* (2014). We also place a batch normalization layer after each convolutional layer and the first fully connected layer. Leaky ReLU non-linearities are used as the activations for all layers except the last fully connected layer in the network. This architecture is used for all the image classifiers in our study.

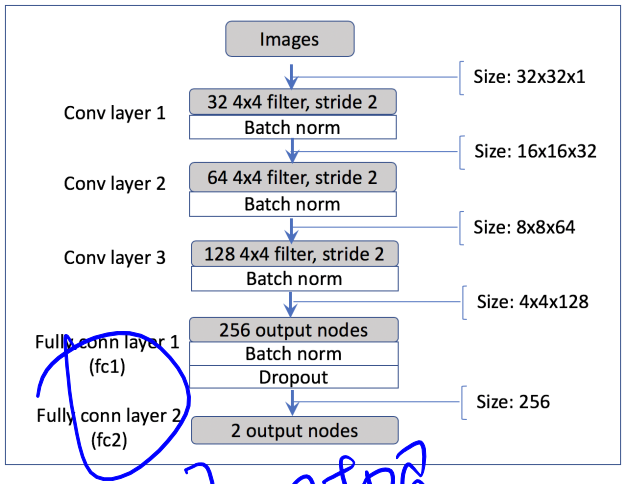


Fig. 2. Neural network structure of image classifier

2.3 Generative adversarial networks

We present a way to use a sizable unlabeled dataset and transfer learning techniques to improve the training of the CNN. Some efforts use pre-trained networks and fine-tune them with the small labeled dataset. However, most publicly available networks are pre-trained on benchmark computer vision datasets, such as ImageNet (Deng *et al.* (2009)). Due to the significant differences between the 3D live images and the image in ImageNet, features learned from computer vision datasets are not directly suitable for our scientific dataset. So we take a different approach that use a generative adversarial networks (GAN) and the sizable unlabeled dataset to learn common features of these images first and then transfer learned features to new CNNs that are further tuned with a small labeled dataset.

Generative adversarial networks (GANs) (Goodfellow *et al.* (2014)) is a generative framework that consists of two competing networks: a generator network and a discriminator network. We use a particular form of GAN, called Wasserstein GAN, in which, the generator network and the discriminator network adopt the same network structure shown in Fig. 2.

Within the GAN framework, the generator produces synthetic data to fool the discriminator while the discriminator network discriminates between real data and synthetic data. The game between the generator G and the discriminator D is the minimax objective:

$$\min_G \max_D E_{x\sim P_{data}}[logD(x)] + E_{\tilde{x}\sim P_g}[log(1-D(\tilde{x}))]. \quad (1)$$

where P_{data} is the distribution of real data and P_g is the distribution of generated data of G defined by $\tilde{x} = G(z), z \sim P_z$. z is the sample from noise distribution P_z, such as the uniform distribution or Gaussian distribution, which is fed to network G as input.

For each update of generator parameters, if the discriminator is trained to optimal, then minimizing the objective function is actually minimizing the Jensen-Shannon (JS) divergence between the real data distribution P_{data} and generated data distribution P_g. However, Arjovsky and Bottou (2017) showed that the JS divergence may not be continuous w.r.t generator parameters, so that training of GAN may be hard to converge. To overcome training difficulty, the Wasserstein distance, which is continuous everywhere and differentiable almost everywhere under mild consumption, is proposed to replace JS divergence (Arjovsky *et al.* (2017)). Wasserstein distance is also referred to as Earth-Mover distance as it shows the minimum effort to transform one distribution into another.

By using the Kantorovich-Rubinstein duality Villani (2009) and a gradient penalty term Gulrajani *et al.* (2017), the cost function of Wasserstein GAN (WGAN) can be write as:

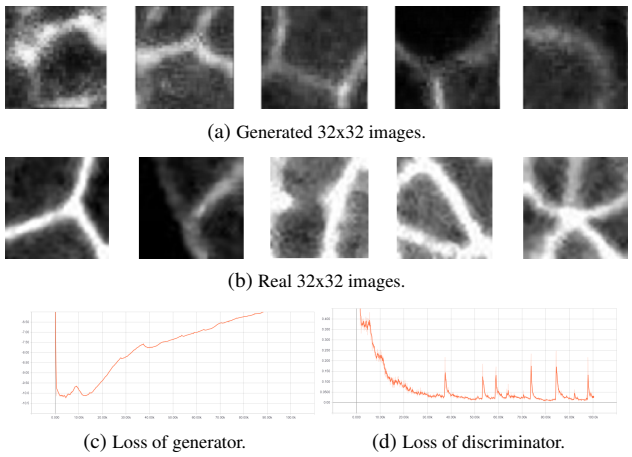
$$\min_G \max_{D\in C} E_{x\sim P_{data}}[D(x)] - E_{\tilde{x}\sim P_g}[D(\tilde{x})] + R. \quad (2)$$

where C is the set of 1-Lipschitz functions.

We adopt cost function Eqn. 2 for the WGAN used in our experiments. We show some samples of generated images patches in Fig. 3(a), and then compare them with image patches in real dataset shown in Fig. 3(b). It is shown that the newly generated images in Fig. 3(a) captured the majority of common features of these live images. The Wasserstein losses for both the generator and discriminator of the 32x32 image case are also shown in Fig. 3(c) and 3(d), respectively.

2.4 GAN-based feature transfer

We first train the GANs with the unlabeled dataset so that the network can learn the majority of common features of the underlying images. Then, we transfer these features to new CNNs and further train the new CNNs with labeled images to capture the target cell structures.



(a) Generated 32x32 images.

(b) Real 32x32 images.

(c) Loss of generator.

(d) Loss of discriminator.

Fig. 3. Generated image patches compared to real image patches and associated Wasserstein losses

Technically, we are interested in the features learned by the GAN discriminator. We create new neural networks, using the same network structure of discriminator without the last fully connected layer, to capture the learned features from the GAN discriminator. The output of the last convolutional layer contains all the structural features learned by a network. The fully-connected layers contain most of the weights in the architecture and a large number of parameters that contain useful information for the target task.

We remove the last layer in the GAN discriminator that is designed to differentiate the difference between real image and generated images. Then we add a new fully-connected layer to classify an image with or without rosette. We use the weights of each convolutional layer and the first fully connected layer of the GAN discriminator (e.g. the fc1 layer in the Fig. 2) to initialize the classifier. Because both classifier and GAN discriminator use batch normalization after each convolutional layer and the first fully connected layer, so there is no bias term for these layers as a new feature of "layers(contrib)" package provided by tensorflow (Abadi *et al.* (2015)). Therefore instead of transferring bias term of each layer, we transfer parameters in batch normalization layers.

After initializing the neural network with parameters from the GAN discriminator, we further train the network using a small manually labeled dataset. The overall workflow of the GAN-based classification is illustrated in Fig. 4.

2.5 Data augmentation and hyperparameters

Because of the limited amount of labeled images, we use several techniques to compensate the potential problems associated with small training datasets for image classification and pattern detection. Specifically, we apply dropout during training after the first fully connected layer to eliminate the over-fit problem. Furthermore, we apply several data augmentation techniques to our dataset including randomly flip the image vertically or horizontally, adjust the brightness and the contrast of the image by a random percentage in a certain range. We use a learning rate of 10^{-5} for the training of the network.

2.6 Computational platform

We implement our networks with tensorflow 1.7.1, a publicly available deep learning framework. More specifically, the convolutional network for classifying is built upon tensorflow's Estimator API with convolutional network as the customized model function. The generative adversarial network is implemented with tensorflow's TFGAN framework with

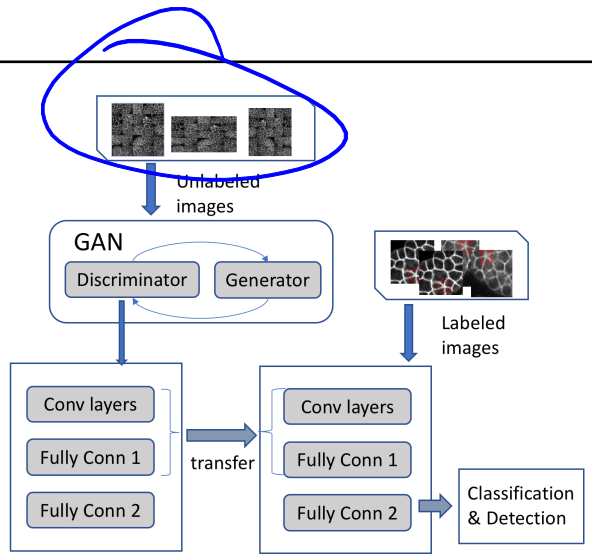


Fig. 4. The overall workflow of GAN-based image classification scheme. There are two steps: common feature learning and target feature learning. An Alex-style network structure shown in Fig. 2 is used for both the GAN framework and the GAN-based classifier

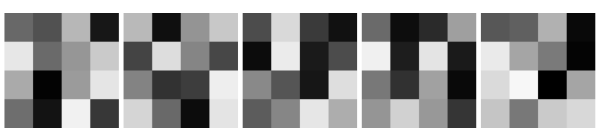
both generator and discriminator are customized. All experiments are performed on an Nvidia DGX server with four cutting-edge Nvidia Tesla V100 GPUs. Each Tesla V100 is equipped with 640 Tensor Cores and 16 GB memory.

3 Results

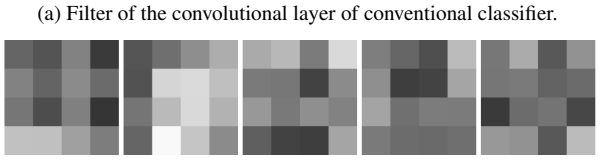
3.1 GAN-based classifier is better trained

It is known that, with a small training dataset, a conventional classifier ran into the data under-fitting problem quickly. [***** need citations here *****]. Compared to a neural network that directly trained on the small labeled dataset, a GAN-based network achieves good testing accuracy and most important, demonstrates a more stable training process. To better understand how the GAN-based classifier works differently from the conventional classifier, we investigate the weights of filters on the first convolutional layer (e.g. the conv1 layer in the Fig. 2). Fig. 5 shows the weights of the filters in first convolutional layer of both a conventional classifier and a GAN-based classifier that are trained for the 32x32 images.

The weights of the GAN-based classifier (shown in 5(b)) is more smooth than weights of the conventional classifier (Fig. 5(a)). Quantitatively, we measured the standard deviations of both sets of weights and it turns out that the STD of the GAN-based classifier is much smaller than that of the conventional classifier (0.192 vs. 0.285). The filter smoothness indicatse that the GAN-based classifier is better trained.



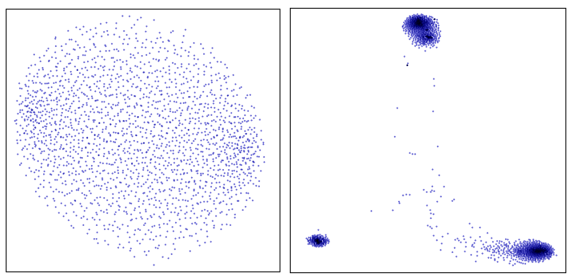
(a) Filter of the convolutional layer of conventional classifier.



(b) Filter of the convolutional layer of GAN-based classifier.

Fig. 5. Visualization of the filters of conventional classifier and GAN-based classifier.

We also analyzed the weights of second last fully-connected (FC) layer (e.g. the fc1 layer in Fig. 2) of both conventional classifier and the GAN-based classifier. There are 2048x256 parameters in the FC layer within the network for 32x32 images. The weights of FC layer contain essential information on how the neural network handle the input images for the classification. Due to the complexity of a convotutional neural network, it is difficult to explicitly explain the role of these weights using regular mapping function (see more information at http://cs231n.github.io/understanding-cnn/). However, we can have some high level estimations on the differences between the conventional and GAN-based classifiers. We use the t-Distributed Stochastic Neighbor Embedding (t-SNE) method (Maaten and Hinton, 2008) to visualize the weights of these networks and explore the the local similarity of the weights from these two networks. The t-SNE visualization result of the weight of the FC layer in the neural network for 32x32 images is illustrated in the Fig. 6.



(a) Weights of the FC layer of a conventional classifier.

(b) Weights of the FC layer of a GAN-based classifier.

Fig. 6. Visualization of the weights of FC layers in a conventional and a GAN-based classifier.

As shown in the Fig. 6, the weights of FC layer in the conventional classifier (illustrated in Fig. 6(a)) is more uniformly distributed and that means the similarity of individual weights is not significant. Compared to Fig. 6(a), Fig. 6(b) has three tight clusters which infer the fact that the GAN-based neural network is more sensitive to the subtle differences among specific structural features, such as the size of structural center and the contrast of edges.

We also compared the visualizations of the output of each layer from both of the GAN-based model and the conventional model trained for the 32x32 images the small dataset in Fig. 7. Since the values of the activations in each layer are not in the same scale and size. We first normalize all feature maps and then reshape them to a size of 32x32.It turns out that in the GAN-based model, in the visualizations of conv layer 2, we saw a clear pattern of rosette, which cannot be found in the conventional classifier.

From the visualizations we also find that the activation maps of the GAN-based model looks brighter than those of the conventional model, especially in Layer 2 and 3. The magnitude of activations (the values the feature maps) is an reasonable indicator that indicates how well a model is trained. These activations, working as feature detectors, with higher values are often more important for the classification task than those with lower values (Molchanov *et al.* (2016); Li *et al.* (2016)). To measure the magnitude of activations quantitatively, we calculated the mean activations in each trained model using a small 32x32 datasets. We compare the mean activations at four different configurations of the training process: (1) random initialization without any training, (2) the initialization with the parameters from the discriminator of the pre-trained GAN but without any fine-tune, (3) the trained conventional model, and (4) the trained GAN-based model (Fig. 8). We find that the mean activation value of each layer from a model without training is quite low. On the other

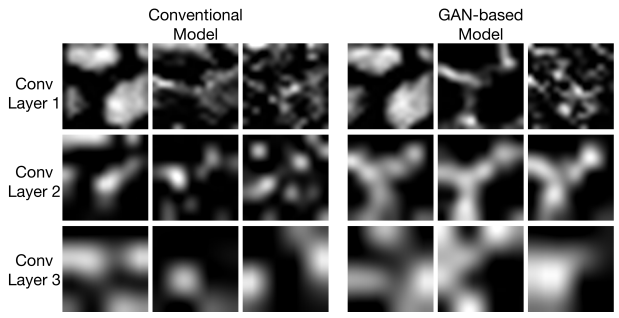


Fig. 7. Visualizations of the output of each layer in GAN-based model vs. a conventional neural network with small training dataset.

hand, those values from the model with the initialization by the pre-trained GAN significantly increase, indicating a much better start point for the fine-tuning on the labeled dataset. We then trained both models for 20k iterations **[***** what kind of training dataset were you used *****]** and it turns out that the mean activations in the GAN-based model is larger than those in the conventional model.

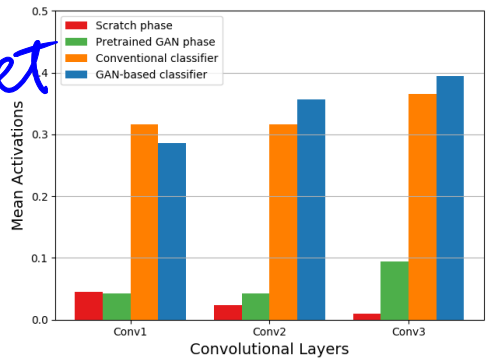


Fig. 8. Mean activations of each convolutional layer inside different configurations with small training samples.

3.2 GAN-based classifier outperformance conventional classifier

3.2.1 Experiments

We first demonstrate the effect of size of training dataset on the performance of conventional and GAN-based classifiers. Then we further evaluate the performance GAN-based classifier with a different size of training datasets.

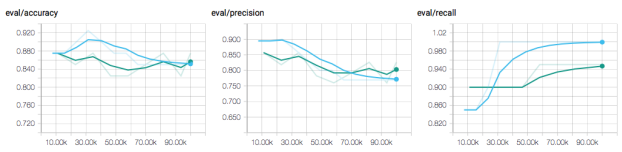
Two sets of experiments are designed to investigate the effect of size of training dataset on the performance on conventional and GAN-based classifiers. We have 78 32x32 rosette images. As rosette images are less frequently appear than non-rosette images in the observation dataset, we selected 3 times more non-rosette images (195 images), from these unlabelled datasets. Since these celegans images are collected in three different dates, we select the annotated images (20 rosette images and 20 non-rosette images, the ratio of rosette to non-rosette is 1:1) collected in one specific day as the validation dataset, all the images from the other two days are used as training dataset (58 rosette images and 175 non-rosette images, the ratio of rosette to non-rosette is around 1:3).

In the first set of experiments, we used the entire training dataset (233 images) and test dataset (40 images). In the second set of experiments, we only used around 1/5 of the training datasets (12 rosette image and 40 non-rosette images) to train neural networks. We adopted all with data augmentation techniques mentioned in Section 2.5 in each experiment.

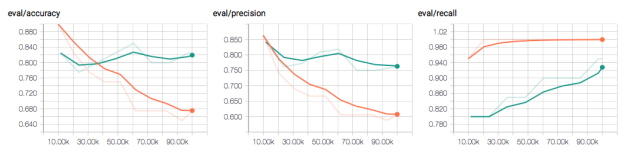
We adopted the F1 score as a measure of our network testing accuracy. F1 score considers both the precision and the recall of the neural network outputs to compute the harmonic average of the precision and recall. An F1 score reaches its best value at 1 (perfect precision and recall) and worst at 0. It is a good measure for our study because the rosette and non-rosette images have a unbalanced distribution within our datasets.

3.2.2 Performance

The model performances of the two sets of experiments are shown in Figure 9.The upper graphs (Fig. 9(a)) show the model performance with the entire training dataset, while Fig 9(b) show the performance of the second set of experiments using 1/5 of training dataset (12 rosette and 40 non-rosette images)



(a) Experiment result and performance comparison with large training dataset (32x32 images). The performance results of conventional classifier are shown in light blue, while the results of GAN-based classifier are shown in green.



(b) Experiment result and performance comparison with small training dataset (32x32 images). The performance results of conventional classifier are shown in orange, while the results of GAN-based classifier are shown in green.

Fig. 9. The comparison of model performance with entire (upper graphs) and 1/5 of the training dataset. The left graphs show the total accuracy of both rosette and non-rosette image prediction. While the middle and right graphs show the model prediction and recall rate of rosette data only.

As shown in the left graphs in the Fig. 9(a), the GAN-based network performs equivalent as the conventional classifier with the entire dataset. The prediction accuracy over the 52 32x32 images is around 86%.

The left (accuracy) graph in the Fig. 9(b) show that the GAN-based classifier outperforms the conventional classifier to overcome the data underfitting problem. The decrease of the precision of conventional classifier prediction (shown in the middle graphs in Fig. 9(b)) is the main reason for performance deterioration. It is also worth mention that the accuracy of the GAN-based classifier works pretty well using the small training datasets is around 82%, comparable to that with the entire training dataset (shown in Fig. 9(a)).

3.3 Dataset size impacts the GAN-based classifier

We further investigated the impact of dataset sizes on the GAN-based classifier. We conducted 11 sets of experiments using whole or partial training datasets (100%, 90%, 80%, 60%, 50%, 40%, 30% 20%, 10% and 2%). In each of the experiments, we conducted 20 individual runs, each used either the entire dataset or a randomly generated partial training dataset. The result is illustrated in Fig 10.

As shown in the Fig. 10, the averaged accuracy of the GAN classifier with the entire dataset is around 84% with a devidation of 4%. The model

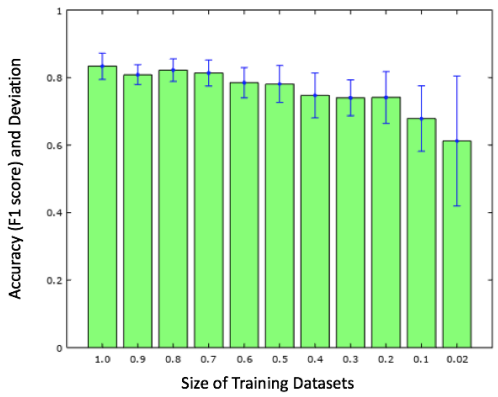


Fig. 10. The accuracy and standard deviation of GAN-based classifier with datasets of different sizes.

accuracy is around 80% (with a standard deviation of 4.5%) when we use more than 40% of the entire training set. The accuracy decreases to 76% and the standard deviation increases to 7% when only 20% of training dataset are used (i.e., 12 rosette images and 40 non-rosette images). When the training dataset is too small (10% or less of the dataset), the accuracy drops much faster and the standard deviations become much bigger. Further evaluations over the precision and recall of these experiments (when over 20% training datasets are used) reveals that the accuracy decrease is mainly due to the deterioration of the precision, that is when the training data is not sufficient, the classifier identify many rosette images but some of predicted images are non-rosettes when compared to the training labels. In a summary, Fig. 10 shows that the GAN-based classifier can deliver comparable accuracy (over 90% of the accuracy with the entire dataset) using around 1/5 of training datasets. This results can potentially save significant time in training data annotation and preparation.

3.4 Rosette detection

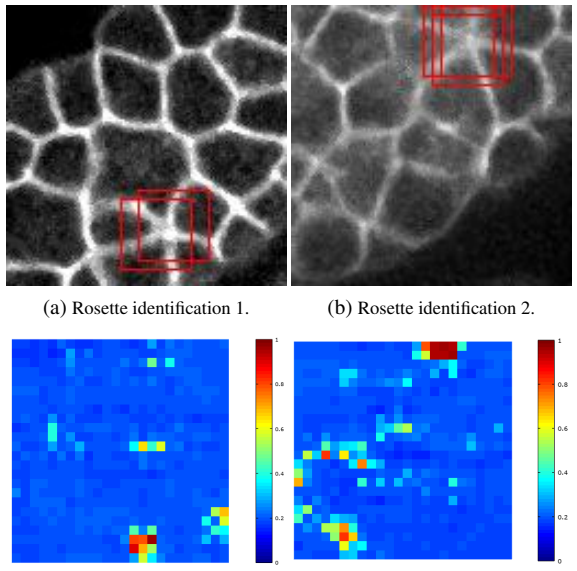
Beside image classification, we are also interested in using a neural network to identify and detect the individual structural pattern inside observation images. We use a sliding window (32x32) approach to identify and detect a 32x32 block of the 128x128 images. We adopt a stride of 4 pixels, therefore, there were 24x24 steps for each 128x128 images. At each step, if the model predicted a higher probability value than a predefined threshold (such as 0.9), we then draw a red-box around current image block. We record all the probability output of each scan and generate a probability heatmap to illustrate the results of rosette identification and detection. Two examples of the results are showed in Fig. 11.

3.5 Rosette dataset

We created an annotated dataset that includes around 400 128*128 images with explicitly marked 32x32 rosette structure(s). We further group the rosette images into two categories according to the probability value from the neural network. There are 85 images identified with a probability value larger than 0.9 and around 300 images with a probability value between 0.80 and 0.9. Examples of these images are illustrated in Fig. 12.

4 Conclusion

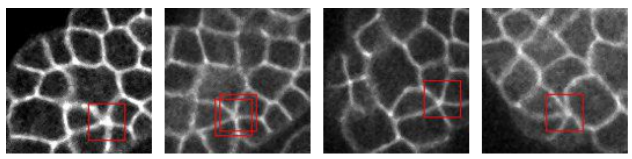
Comparing to the multi-institutional efforts for large-scale data exploration Rajkomar *et al.* (2018), to establish a sufficient training dataset from a limited collection of microscopy images is a challenging but critical step to enable deep learning based pattern identification. We presented a deep learning approach to efficiently classify images with a particular cellular structure with relative small unlabeled images and minimal annotated samples. The approach has two steps: first we used 11250 unlabeled image and a discriminator network inside a GAN framework to capture common features of microscope images, then we transfer the discriminator networking into a new classifier for a target image classification with couple dozens manually-annotated training samples. Considering the data limitation, we adopted an Alex-style network structure, instead of a more complex network structure, for the image classification. We think the methodology and concepts can be applied to other groups who are interested in deep learning for identifying and detecting unique structures within the microscopic data, such as fly, mice and human brains.



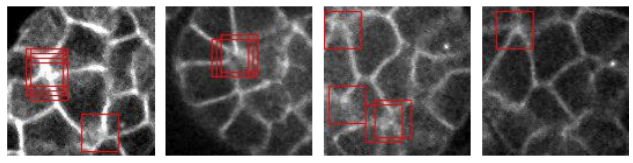
(a) Rosette identification 1. (b) Rosette identification 2.

(c) Probability heatmap of rosette identification 1. (d) Probability heatmap of rosette identification 2.

Fig. 11. Example rosette images and associated probability heatmap for rosette detection.



(a) Example of rosette images (with a high level of confidence).



(b) Example of rosette images (with a moderate level of confidence).

Fig. 12. Rosette detection with high (upper graph) and medium probability (lower graph).

5 Data and software availability

We created an annotated dataset that includes 128x128 images with explicitly marked 32x32 rosette structure(s). The dataset is available in dropbox (www.dropbox.com/sh/vlz3m2uzw73svts/AADHQpVGKNwnEskRF21oGJKTa?dl=0). These images can serve

as a benchmark training dataset for further algorithm and application improvements. Related code and software utilities are located at https://github.com/DongCiLu/BioGAN.

Funding

This study is supported by an NIH research project grants (R01GM097576). Research in the Bao lab is also supported by an NIH center grant to MSKCC (P30CA008748).

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Arjovsky, M. and Bottou, L. (2017). Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862*.

Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein gan. *arXiv preprint arXiv:1701.07875*.

Bao, Z., Murray, J. I., Boyle, T., Ooi, S. L., Sandel, M. J., and Waterston, R. H. (2006). Automated cell lineage tracing in caenorhabditis elegans. *Proceedings of the National Academy of Sciences of the United States of America*, **103**(8), 2707–2712.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE.

Doersch, C., Gupta, A., and Efros, A. A. (2015). Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1422–1430.

Giurumescu, C. A., Kang, S., Planchon, T. A., Betzig, E., Bloomekatz, J., Yelon, D., Cosman, P., and Chisholm, A. D. (2012). Quantitative semi-automated analysis of morphogenesis with single-cell resolution in complex embryos. *Development*, **139**(22), 4271–4279.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. (2017). Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pages 5769–5779.

Jones, T. R., Carpenter, A. E., Lamprecht, M. R., Moffat, J., Silver, S. J., Grenier, J. K., Castoreno, A. B., Eggert, U. S., Root, D. E., Golland, P., and Sabatini, D. M. (2009). Scoring diverse cellular morphologies in image-based screens with iterative feedback and machine learning. *Proceedings of the National Academy of Sciences*, **106**(6), 1826–1831.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Kyoda, K., Adachi, E., Masuda, E., Nagai, Y., Suzuki, Y., Oguro, T., Urai, M., Arai, R., Furukawa, M., Shimada, K., *et al.* (2012). Wddd: worm developmental dynamics database. *Nucleic acids research*, **41**(D1), D732–D737.

Li, C., Wang, Z., and Qi, H. (2018). Fast-converging conditional generative adversarial networks for image synthesis. *arXiv preprint arXiv:1805.01972*.

Li, H., Kadav, A., Durdanovic, I., Samet, H., and Graf, H. P. (2016). Pruning filters for efficient convnets. *arXiv preprint arXiv:1608.08710*.

Maaten, L. v. d. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, **9**(Nov), 2579–2605.

Molchanov, P., Tyree, S., Karras, T., Aila, T., and Kautz, J. (2016). Pruning convolutional neural networks for resource efficient inference. *arXiv preprint arXiv:1611.06440*.

Noroozi, M. and Favaro, P. (2016). Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pages 69–84. Springer.

Odena, A., Olah, C., and Shlens, J. (2016). Conditional image synthesis with auxiliary classifier gans. *arXiv preprint arXiv:1610.09585*.

Pan, S. J., Yang, Q., *et al.* (2010). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, **22**(10), 1345–1359.

Radford, A., Metz, L., and Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*.

Rajkomar, A., Oren, E., Chen, K., Dai, A. M., Hajaj, N., Hardt, M., Liu, P. J., Liu, X., Marcus, J., Sun, M., *et al.* (2018). Scalable and accurate deep learning with electronic health records. *npj Digital Medicine*, **1**(1), 18.

Schneider, C. A., Rasband, W. S., and Eliceiri, K. W. (2012). Nih image to imagej: 25 years of image analysis. *Nature methods*, **9**(7), 671.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. (2014). Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S. E., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 1–9.

Villani, C. (2009). Optimal transport, old and new. grundlehren der mathematischen wissenshaften 338.

Wang, Z., Ramsey, B. J., Wang, D., Wong, K., Li, H., Wang, E., and Bao, Z. (2016). An observation-driven agent-based modeling and analysis framework for c. elegans embryogenesis. *PloS one*, **11**(11), e0166551.

Wang, Z., Wang, D., Li, H., and Bao, Z. (2017). Cell neighbor determination in the metazoan embryo system. In *Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, pages 305–312. ACM.

Wang, Z., Wang, D., Li, C., Xu, Y., Li, H., and Bao, Z. (2018). Deep reinforcement learning of cell movement in the early stage of c. elegans embryogenesis. *Bioinformatics*, **1**, 9.

Weigert, M., Schmidt, U., Boothe, T., Andreas, M., Dibrov, A., Jain, A., Wilhelm, B., Schmidt, D., Broaddus, C., Culley, S., *et al.* (2017). Content-aware image restoration: Pushing the limits of fluorescence microscopy. *bioRxiv*, page 236463.