Toward Automatic Phenotyping of Developing Embryos from Videos

Feng Ning, Damien Delhomme, Yann LeCun,
Fabio Piano, Léon Bottou, Paolo Emilio Barbano

EDICS categories: 2-SEGM, 2-NEUR, 2-NFLT

***Abstract*— We describe a trainable system for analyzing videos of developing C. elegans embryos. The system automatically detects, segments, and locates cells and nuclei in microscopic images. The system was designed as the central component of a fully-automated phenotyping system. The system contains three modules (1) a convolutional network trained to classify each pixel into five categories: cell wall, cytoplasm, nucleus membrane, nucleus, outside medium; (2) an Energy-Based Model which cleans up the output of the convolutional network by learning local consistency constraints that must be satisfied by label images; (3) A set of elastic models of the embryo at various stages of development that are matched to the label images.**

***Index Terms*— image segmentation, convolutional networks, nonlinear filter, energy-based model**

I. Introduction

A. Automatic Phenotyping

One of the major goals of biological research in the post-genomic era is to characterize the function of every gene in the genome. One particularly important subject is the study of genes that control the early development of animal embryos. Such studies often consist in knocking down one or several genes and observing the effect on the developing embryo, a process called *phenotyping*.

As an animal model, the nematode *C.elegans* is one of the most amenable to such genetic analysis because of its short generation time, small genome size, and availability of a rapid gene knock-down approach, RNAi(RNA interface) [7].

Since the completion of the *C.elegans* genome sequence and identification of its roughly $20,000$ protein-coding genes in 1998 [6], extensive research has been done on analyzing how these genes function *in vivo*. Early embryonic events provide a good model to assess specific roles genes play in a developmental context. Early *C.elegans* embryos are easily observable under a microscope fitted with Nomarski Differential Interference Contrast (DIC) optics. When observing normal (*wild type*) embryos it is possible to visualize important cellular functions such as nuclear movements and fusions, cytokinesis and the setting up of crucial cell-cell contacts. These events are highly reproducible from embryo to embryo and deviate from normal behaviors when the function of a specific gene is depleted [10] [30] [31] [40], allowing the association of a gene's activity with specific early embryonic events.

A typical experiment consists in knocking down a gene (or a set of genes), and recording a time-lapse movie of the developing embryo through DIC microscopy. Figure 1 shows a few frames extracted from the movie of a normally developing *C. elegans* embryo from the fusion of the pronuclei to the four-cell stage.

Using RNAi, several research groups have gathered a large collection of such movies. Many of these movies depict cellular behaviors in the early embryos that deviate from the wild type, and some show dramatic problems during embryonic development. Although initial analyses of the movies have been performed by hand, automating the analysis of the cellular behaviors would augment our ability to process the large amounts of data being currently produced, and could reveal more subtle quantitative defects that cannot be detected by manual analysis.

One important classification task is to automatically detect whether the development is normal (and therefore, not particularly interesting), or abnormal and worth investigating. Another important task is to automatically extract quantitative measurements such as the number of cells, the relative positions of the cell nuclei, the time between each cell division, etc... Ultimately, one may want an automated system for classifying the movies into a number of known scenarios of normal or abnormal development.

This paper focuses on the detection, identification, and measurement of various objects of interests in the embryo, namely the cell nuclei, cytoplasms, and cell walls, from sequences of images obtained through DIC microscopy. The system described here is the key component of a fully automated analysis system under development. The present study primarily concerns the early stages of development, from fertilization to the four-cell stage.

Although the development of *C. elegans* embryos is the subject of numerous studies from biologists, there have been very few attempts to automate the task of analyzing DIC image sequences. The most notable exception is the work of Yasuda

Feng Ning is with the Courant Institute of Mathematical Sciences, New York University. Room 1006, 715 Broadway, New York, NY 10003. Email: fengning@cs.nyu.edu.

Damien Delhomme is with Naskeo, Ecole Centrale de Paris, 92295 Chatenay-Malabry, France. Email: damien.delhomme@envalia.com.

Yann LeCun is with the Courant Institute of Mathematical Sciences, New York University. Room 1220, 715 Broadway, New York, NY 10003. Email: yann@cs.nyu.edu.

Fabio Piano is with Biology Department, New York University. 1009 Silver Center, New York, NY 10003. Email: fp1@nyu.edu.

Léon Bottou is with NEC Labs-America, 4 Independency Way, Princeton, NJ 08540. Email: leon@bottou.org.

Paolo Barbano is with the Department of Applied Mathematics, Yale University and Courant Institute of Mathematical Sciences, New York University. 10 Hillhouse Ave, New Haven, CT 06520. Email: paoloemilio.barbano@math.yale.edu

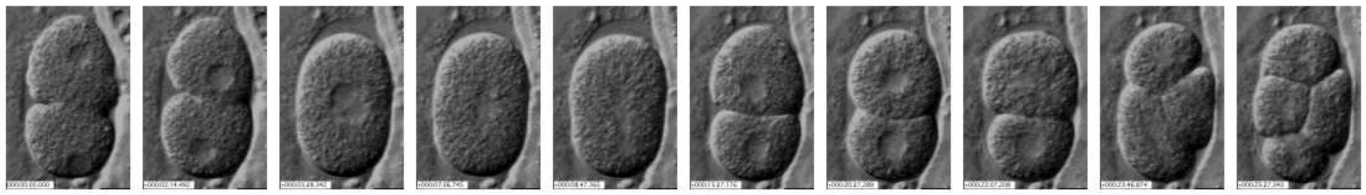


Fig. 1. Snapshots of the early development stages of a wild type C.elegans embryo obtained through DIC microscopy.

et al. [39], which describes a computer vision approach to the detection the nuclei and cell walls. Their method is based on the combination of several types of edge features. Because DIC microscopy images are very noisy and anisotropic, the method produces a large number of false positives (e.g. areas falsely detected as cell nuclei) that must be manually corrected. One conclusion from this work is that DIC images are not easily analyzed with commonly-used feature detectors. In this paper, we propose to rely on machine learning methods to produce a more reliable image segmentation system.

Learning methods have been used for low-level image processing and segmentation with some success over the last few years. A notable example is the object boundary detection system of Martin et al. [25], [24]. Closer to our application is the detection and classification of sub-cellular structures in fluorescence microscopy images. Machine learning and adaptive pattern recognition methods have been widely applied to this problem in a series of influential work [23], [12]. These systems rely on the time-honored method of extracting a large number of carefully engineered features, while using learning methods to select and exploit these features.

B. Overview of the system

The method proposed in this paper consists in learning the entire processing chain *from end to end*, from raw pixels to ultimate object categories. The system is composed of three main modules.

The first module is a trainable *Convolutional Network*, which labels each pixel in a frame into one of five categories. The categories are: cell nucleus, nuclear membrane, cytoplasm, cell wall, and outside medium. The main advantage of Convolutional Nets is that they can learn to map raw pixel images into output labels, synthesizing appropriate intermediate features along the way, and eliminating the need for manually engineered features. They have been widely applied to detection and recognition tasks such as handwriting recognition with integrated segmentation (see [19] for a review), hand tracking [27], face recognition [18], face detection [34], [8], [28], and generic object recognition [11]. The main advantages of Convolutional Networks is that they can operate directly on raw images

The architecture of the convolutional network is designed so that each label can be viewed as being produced by a non-linear filter applied to a 40×40 pixel window centered on the pixel of interest in the input image. This convolutional network is trained in supervised mode from a set of manually labeled images. The five categories may appear somewhat redundant: it would be sufficient to label the nucleus, cytoplasm, and external medium to locate the nuclear membrane and the cell wall. However, including the boundaries as explicit categories introduces redundancy in the label images that can be checked for consistency.

Ensuring local consistency is the role of the next module. Since the label of each pixel is produced independently of the labels of neighboring pixels, the predicted label image may indeed contain local inconsistencies. For example, an isolated pixel in the outside medium may be erroneously classified as nucleus. Since nucleus pixels must be surrounded by other nucleus pixels or by nuclear membrane pixels, it would seem possible to clean up the label image by enforcing a set of local consistency constraints. To implement this process, we used an *energy based model* (EBM) [33], [16], [21]. EBMs are somewhat similar to Markov Random Fields, and can be seen as a sort of non-probabilistic Conditional Random Field [17]. The EBM used in the present system can be viewed a scalar-valued "energy" function $E(f(X), Y)$, where $f(X)$ is the label image produced by the convolutional net, and Y is the cleaned-up image. The EBM is trained so that when $f(X)$ is a predicted label image and Y is the corresponding "correct" (cleaned-up) label image, the energy $E(f(X), Y)$ will be smaller than for any other ("incorrect") value of Y. The cleanup process consists in searching for the Y that minimizes $E(f(X), Y)$ for a given $f(X)$. This approach is related to the relaxation labeling method [13]. While learning methods have been used to estimate the coupling coefficients in relaxation labeling systems [29], the method used here is based on minimizing a new type of contrastive loss function [21].

The third component of the system models the embryos and their internal parts by matching deformable templates to the label images. This module is used to precisely locate and count parts such as cells nuclei, and cell walls. It is also used to determine the stage of development of the embryo in the image. This technique is related to the classical active contour method [14], [26], and very similar to elastic matching methods based on the Expectation-Maximization algorithm as described in [32], [3].

The following sections describe the three modules of the system in detail.

II. Convolutional network

A Convolutional Network is a trainable system whose architecture is specifically designed to handle images or other 1D or 2D signals with strong local correlations. A Convolutional Network can be seen as a cascade of multiple non-linear local filters whose coefficients are learned to optimize an overall performance measure. Convolutional Networks have been applied with success to a wide range of applications [19], [27], [18], [34], [8], [28], [11].

Convolutional Networks are specifically designed to handle the variability of 2D shapes. They use a succession of layers of trainable convolutions and spatial subsampling interspersed with sigmoidal non-linearities to extract features with increasingly large receptive fields, increasing complexity, and increasing robustness to irrelevant variabilities of the inputs. The convolutional net used for the experiments described in this paper is shown in figure 2.

Each convolutional layer is composed of a set of planes called *feature maps*. The value at position (x, y) in the j-th feature map of layer i is denoted c_{ijxy}. This value is computed by applying a series of convolution kernels w_{ijk} to feature maps in the previous layer (with index $i-1$), and passing the result through a sigmoid function. The width and height of the convolution kernels in layer i are denoted P_i and Q_i respectively. In our network, the kernel sizes are between 2 and 7. More formally, c_{ijxy} is computed as:

$$c_{ijxy} = \tanh\left(b_{ij} + \sum_k \sum_{p=0}^{P_i-1} \sum_{q=0}^{Q_i-1} w_{ijkpq} c_{(i-1),k,(x+p),(y+q)}\right) \tag{1}$$

where p, q index elements of the kernel w_{ijk}, $\tanh$ is the hyperbolic tangent function, i is the layer index, j is the index of the feature map within the layer, k indexes feature maps in the previous layer, and b_{ij} is a bias. Each feature map is therefore the result of a sum of discrete convolutions of the previous layer maps with small-size kernels, followed by a point-wise squashing function. The parameters w_{ijkpq} and b_{ij} are all subject to learning.

Subsampling layers have the same number of feature maps as the convolutional layer that precedes them. Each value in a subsampling map is the average of the values in a 2×2 neighborhood in the corresponding feature map in the previous layer. That average is added to a trainable bias, multiplied by a trainable coefficient, and the result is passed through the $\tanh$ function. The 2×2 windows are stepped without overlap. Therefore the maps of a subsampling layer are one half the resolution of the maps in the previous layer. The role of the subsampling layers is to make the system robust to small variations of the location of distinctive features.

Figure 2 only shows a portion of the network: the smallest portion necessary to produce a single output label. Each output is influenced by a 40×40 pixel window on the input. The full network can be seen as multiple replicas of this network applied to all 40×40 windows stepped every 4 pixels on the input image (more on this later). The window size was chosen so that the system would have enough context information to make an informed decision about the category of a pixel. For example, the local texture in the nucleus region is often indistinguishable from that of the external medium. Therefore, distinguishing nucleus pixels from external medium pixels can only be performed by checking if the pixel is within a roughly circular region surrounded by cytoplasm. Since the nuclei are typically less than 40 pixels in diameter, we set the window size to 40×40 to ensure that at least some of the nuclear membrane and the cytoplasm will be present in every window containing nucleus pixels. Once the input window size is chosen, the choice of the kernel size and subsmpling ratio for each layer is quite constrained. The first layer (marked C1) contains 6 feature maps with 7×7 pixel convolution kernel. The second layer (S2), is a subsampling layer with 2×2 subsampling ratios. The third layer (C3) uses 6×6 convolution kernels. Each of the 16 maps in C3 combines data from several maps in S2 by applying a separate convolution kernel to each map, adding the results, and applying the sigmoid. Each feature variable in C3 is influenced by an 18×18 pixel window on the input. Each C3 map combines input from a different subset of of S2 maps, with a total of 61 individual kernels. Layer S4 is similar to S2 and subsamples C3 by a factor of 2. Layer C5 comprises 40 feature maps that use 6×6 convolution kernels. There is one kernel for each pair of feature map in S4 and C5. The output layer contains five units, one for each category.

One key advantage of convolutional nets is that they can be applied to images of variable size. Applying the network to a large image is equivalent (but considerably cheaper computationally) to applying a copy of the single-output network to every 40×40 window in the input stepped every 4 pixel. More precisely, increasing the input size by 4 pixels in one direction will increase the size C1 by 4 pixels, S2 and C3 by 2 pixel, and S4, C5, and the output by 1 pixel. The size of the output in any dimension is therefore $(N - 36)/4$, where N is the size of the input image in that dimension. Consequently, the convolutional net produces a labeling for every 4×4 block of pixels in the input, taking information from a 40×40 window centered on that block of pixels. Figure 2 shows the size of each layer when a 40×40 pixel input is used and a single output vector is produced. Figure 4 shows the result of applying the convolutional network to an image, which produces a label image with $1/4$ the resolution of the input. It would be straightforward to modify the method to produce a label image with the same resolution as the input. However, we determined that the current application did not require pixel-level accuracy.

A. Datasets and Training

Training images were extracted from 5 different movies of *C. elegans* embryos. 10 frames were extracted from each movie, every 10 frames, for a total of 50 frames. Testing images were extracted from a disjoint set of 3 movies (of three different embryos). Similarly, 10 frames were extracted (separated by 10 frames) from each test movie, for a total of 30 frames. The sample frames were picked every 10 frames in the movies so as to have a representative set covering the various stages of embryonic development.

Frames from different movies had different sizes, but were typically around 300×300 pixels. All images were 8-bit grayscale. The movies were stored in Apple Quicktime format, whose compression method introduces some quantization and blocking artifacts in the frames. Working with compressed video make the problem more difficult, but it will allow us to

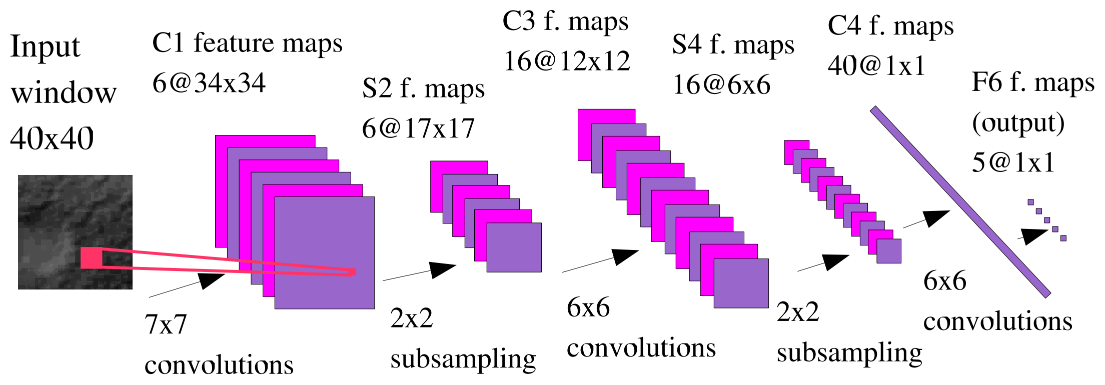


Fig. 2. The convolutional network architecture. The feature map sizes indicated here correspond to a 40×40 pixel input image, which produces a 1×1 pixel output with 5 components each. Applying the network to an $N_x \times N_y$ pixel image will result in output maps of size $[(N_x - 36)/4] \times [(N_y - 36)/4]$.

tap into a larger pool of movies produced by various groups around the world, and distributed in compressed formats.

1) preprocessing: DIC images are not only very noisy, but also very anisotropic. The DIC process creates an embossed "bas relief" look that, while pleasing to the human eye, makes processing the images quite challenging. For example the cell wall in the upper left region of the raw image in figure 4 looks quite different from the cell wall in the lower right region. We decided to design a linear filter that would make the images more isotropic, while preserving the texture information. The linear filter used was equivalent to computing the difference between the image and a suitably shifted version of it. A typical resulting image is shown in figure 4((a), bottom). The pixel intensities were then centered so that each image had zero mean, and scaled so that the standard deviation was 1.0. An unfortunate side effect of this pre-processing is that it makes the quantization artifacts of the video compression more apparent. Better preprocessing will be considered for future embodiments of the system. It should be emphasized that the purpose of this preprocessing is merely to make the image features more isotropic. The purpose is not to recover the optical pathlength, as several authors working with DIC images have done [35].

2) labels: Each training and testing frame was manually labeled with a simple graphical tool by a single person. Labeling the images in a consistent manner is very difficult and tedious. Therefore, we could not expect the manually produced labels to be perfectly consistent. In particular, it is very common for the position nucleus boundary or the cell wall to vary by several pixels from one image to the next.

Consequently, it appeared necessary to use images of desired labels that could incorporate a bit of slack in the position of the boundaries. We used a very simple method which consists in deriving two label images from each human-produced label image. The process is described in figure II-A.2. The first image, called M1, contains no boundary labels. It is obtained by turning all the nuclear membrane pixels into either nucleus or cytoplasm using a simple nearest neighbor rule. The second label image, M2, is obtained by dilating the boundaries by one pixel on each side, thereby producing a 3-pixel wide boundary.

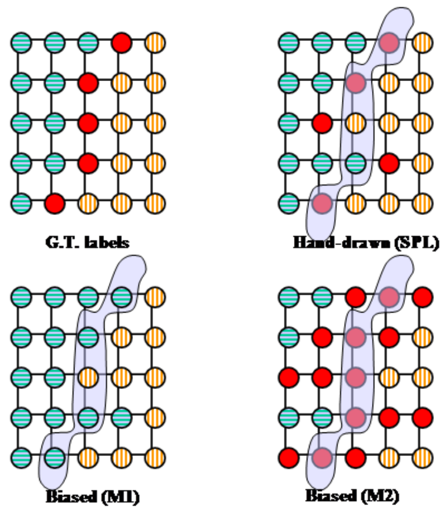


Fig. 3. How the M1 and M2 label images are produced. The ground truth label image (GT) is unattainable. The human-produced labels may contain error and inconsistencies. The M1 label image is derived from the human produced labels by removing all boundary pixels, whereas the M2 label image is produced by making the boundary 3 pixel wide so as to encompass the ground truth.

Two separate networks were trained to produce the two different sets of label images M1 and M2.

3) training set and test set: The simplest way to train the system would be to simply feed a whole image to the system, compare the full predicted label image to the ground truth, and adjust all the network coefficients to reduce the error.

This "whole-image" approach has two deficiencies. First, there are considerable differences between the numbers of pixels belonging to each category. This may cause the infrequent categories to be simply ignored by the learning process. Second, processing a whole image at once can be seen as being equivalent to processing a large number of 40×40 pixel windows in a batch. Previous studies have shown that performing a weight update after each sample leads to faster convergence than updating the weights after accumulating gradients over a batch of samples [20]. Therefore, we chose to break up the training images into a series of overlapping 40×40 windows that can be processed individually. Overall, from the 50 frames in the training set, $190,440$ windows of size 40×40 pixels were extracted. To each such window was associated the desired labels (for M1 and M2) of the central pixel in the window. Each pair of window and label was used as a separate training sample for the convolutional network, which therefore produced a single output vector (a 1 pixel output map). There were wide variations in the number of training samples for each category: 3333 windows were labeled nucleus, 12939 nuclear membrane, 80142 cytoplasm, 39612 cell wall, and 54414 external medium. To correct these wide variations, a class frequency equalization method was used. A full learning epoch through the training set consisted in $272,070 = 5 \times 54414$ pattern presentations. During one epoch, each sample labeled "external medium" was seen once, while samples from the other categories were repeated $54414/P$ times, where P is the number of samples from that category. Therefore each category was presented an equal number of times (54414) during each epoch.

The network was trained to minimize the mean squared error between its output vector and the target vector for the desired category. The target vectors were $[+1, -1, -1, -1, -1]$ for nucleus, $[-1, +1, -1, -1, -1]$ for nucleus membrane, $[-1, -1, +1, -1, -1]$ for cytoplasm, $[-1, -1, -1, +1, -1]$ for cell wall, and $[-1, -1, -1, -1, +1]$ for external medium. The training procedure used a variation of the back-propagation algorithm to compute the gradients of the loss with respect to all the adjustable parameters, and used an "on-line" version of the Levenberg-Marquardt algorithm with a diagonal approximation of the Hessian matrix to update those parameters (details of the procedure can be found in [19]).

Two separate networks were trained, one with the M1 labels and another one with the M2 labels. Results are reported for the network trained with the M2 labels.

B. Results

The network was trained for 6 epochs on the frequency-equalized dataset. The pixel-wise error rate of the network trained with the M2 labels was measured. The pixel-wise error rate on the *non frequency equalized* training set (i.e. on the 50 frames from the training set) was 25.6%. The pixel-wise error rate was 29.0% on the 30 test frames. It must be emphasized that pixel-wise error rate is a bad indicator of the overall system performance. First of all, many errors are isolated points that can easily be cleaned up by post-processing. Second, it is unclear how many of the errors can be attributed to inconsistencies in the human-produced labels, and how many can be attributed to truly inaccurate classifications. Third, and more importantly, the usefulness of the overall system will be determined by how well the cells and nuclei can be detected, located, counted, and measured.

Figure 4 shows a sample image (top left), a pre-processed version of the image (bottom left), and the corresponding internal state and output of the convolutional network. Layers C1, C3, C5 and F6 (output) are shown. The segmented regions of the five categories (nucleus, nuclear membrane, cytoplasm, cell wall, external medium) are clearly delineated in the five output maps.

The labeling produced by the network for several sample images, with a false color scheme to represent the various categories, is shown in figure 5. The essential elements of the embryos are clearly detected. The cell nuclei are correctly labeled before, during, and after the fusion of the pro-nuclei. The cell wall is correctly identified by the M2 network. However, the detection of new cell walls created during cell division (mitosis) seems to be more difficult.

It takes 90 minutes for one iteration of training process (one pass through the frequency-equalized training set), on a Xeon-based workstation running at 2.2 GHz. We train the machine with the M1 labels and the M2 labels in two seperate processes up to 6 epochs, so the total CPU time is approximately 18 hours.

The main advantage of the convolutional network approach is that the low-level features are automatically learned. A somewhat more traditional approach to classification consists in selecting a small number of relevant features from a large set. One popular approach is to automatically generate a very large number of simple "kernel" features, and to select them using the Adaboost learning algorithm [36]. Another popular approach is to build the feature set by hand. This approach has been advocated in [5] for the classification of sub-cellular structures. We believe that these methods are not directly applicable to our problem because the regions are not well characterized by local features, but depend on long-range context (e.g. a nucleus is surrounded by the cytoplasm). This kind of contextual information is not easily encoded into a feature set.

III. Energy-based model

The Convolutional Network gives predictions on a per-pixel basis. While it is trained to produce the best possible labeling, there is no specific mechanism to ensure that elementary consistency constraints of labels within a neighborhood are respected. Some of those local constraints are easy to formulate. For example, a nuclear membrane pixel must be connected to other nuclear membrane pixels, and must have a nucleus pixel on one side, and a cytoplasm pixel on the other side. A popular way to model local consistency constraints in images is to use Graphical Models, particularly Markov Random Field (MRF) [22], which incorporate local 2D interactions between variables.

Traditionally, the interactions terms between the variables in an MRF are encoded by hand. While some of the rules for our application could be encoded by hand, we chose to learn them from data using the Energy-Based Model framework.

Traditionally, MRFs and other Graphical Models are viewed as probabilistic generative models, where each configuration of the input variable is associated with a probability. To ensure proper normalization of the distribution, the integral (or the sum) of that probability over all possible input configurations must be one. Learning the parameters of such a model is generally performed by maximizing the likelihood of the training

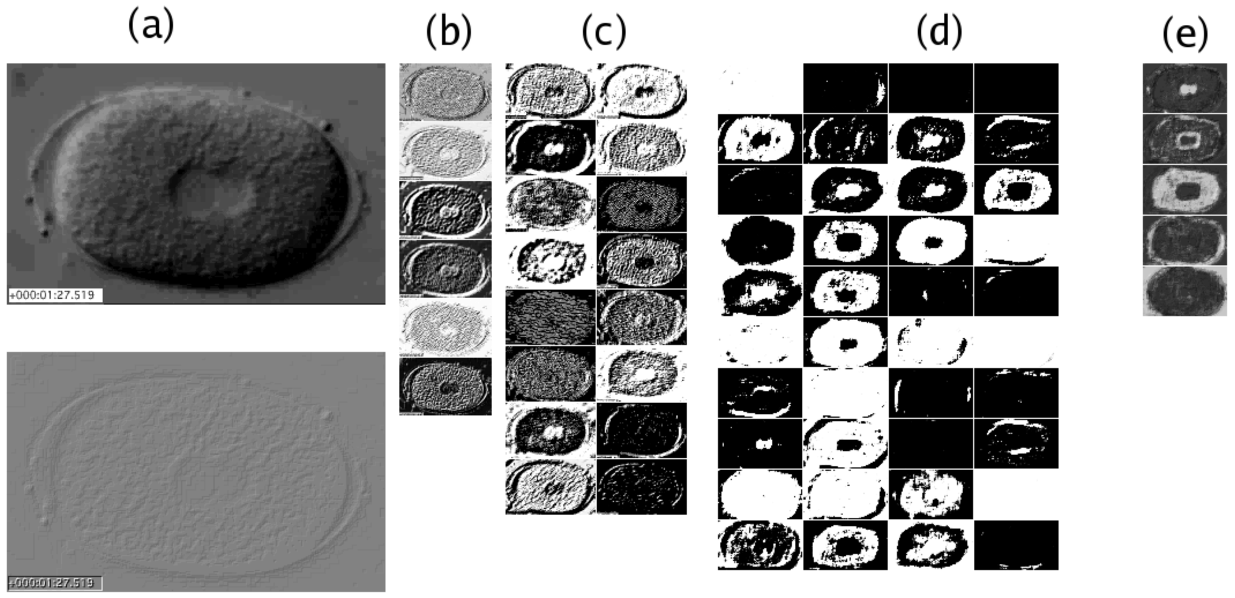


Fig. 4. convolutional network applied to a sample image. (a) top: raw input image; bottom: pre-processed image; (b) state of layer C1; (c) layer C3; (d): layer C5; (e): output layer. The five output maps correspond to the five categories, respectively from top to bottom: nucleus, nucleus membrane, cytoplasm, cell wall, external medium. The properly segmented regions are clearly visible on the output maps.

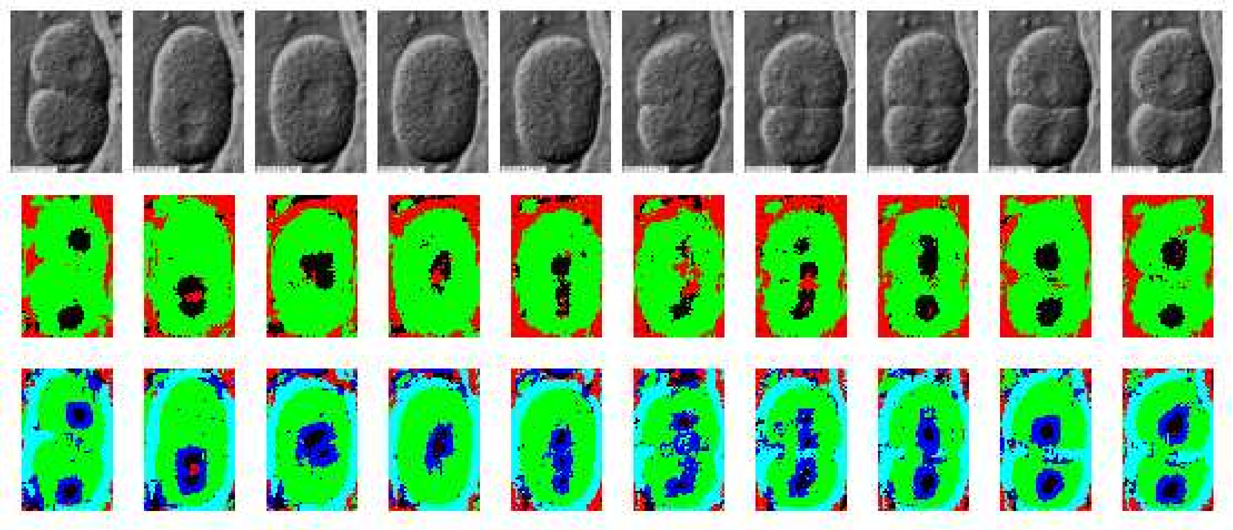


Fig. 5. Pixel labeling produced by the convolutional network . Top to bottom: input images; label images produced by the network trained with M1 labels; label images produced by the M2 network Because each output is influenced by a 40×40 window on the input, no labeling can be produced for pixels less than 20 pixels away from the image boundary.

data under the model. Unfortunately, this often requires that the probability distribution be explictely normalized. This normalization is generally intractable because it entails computing the *partition function*, the normalization term which is a sum over all possible input configurations (all possible label images in our case).

Energy-Based Models [33], [21] associate a scalar *energy* to each configuration of the input variables. Making an inference with an EBM consists in searching for a configuration of the variables to be predicted that minimizes the energy. EBMs have considerable advantages over traditional probabilistic models: (1) There is no need to compute the partition functions that may be intractable; (2) because there is no requirement for normalizability, the repertoire of possible model architectures that can be used is considerably richer than with probabilistic models.

Training an EBM consists in finding values of the trainable parameters that associate *low energies* to “desired” configurations of variables (e.g. observed on a training set), and *high energies* to “undesired” configurations. With properly normalized probabilistic models, increasing the likelihood of a “desired” configuration of variables will automatically decrease the likelihoods of other configurations. With EBMs, this is not the case: making the energy of desired configurations low

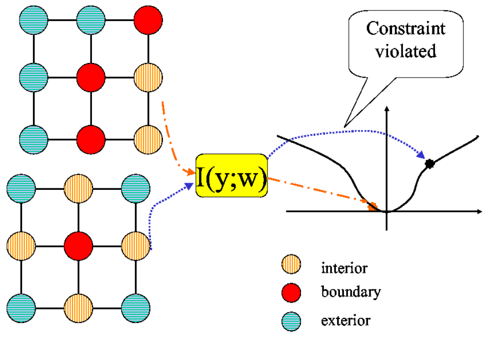


Fig. 6. Local constraints: we illustrate the idea for a 3-category classification problem. The top-left configuration is consistent, so assigned low energy, while the bottom-left configuration is not consistent.

may not necessarily make the energies of other configurations high. Therefore, one must be very careful when designing loss functions for EBMs. We must make sure that the loss function we pick will effectively drive our machine to approach the desired behavior.

A. The architecture of the EBM

The EBM is a scalar function $E(W, Y, f_1(X), f_2(X)))$ where W is the parameter vector to be learned, Y is the label image to be predicted using the EBM, $f_1(X)$ is the label image produced by the M1 convolutional network, and $f_2(X)$ the label image produced by the M2 convolutional network. Each of the variables $Y, f_1(X), f_2(X)$ are 3D arrays of size $5 \times N_x \times N_y$, where N_x and N_y are the dimensions of the label images.

Operating the EBM consists in running the input image through the M1 and M2 convolutional networks, and clamping the $f_1(X)$, and $f_2(X)$ inputs of the EBM to the values thereby produced. Then the Y input is initialized with the value $f_2(X)$, and an optimization algorithm is run to find a value of Y that locally minimizes $E(W, Y, f_1(X), f_2(X)))$. The quantity at each pixel location of Y, $f_1(X)$, and $f_2(X)$ is a discrete variables with 5 possible values: nucleus, nuclear membrane, cytoplasm, cell wall, external medium. A number of Markov-Chain Monte-Carlo (MCMC) methods were tested for minimizing the energy, including simulated annealing with Gibbs sampling. In the end, a simple "greedy" deterministic descent was used. The sites are updated sequentially and set to the configuration that minimizes the energy, keeping the other sites constant.

The EBM is composed of two modules or *factors*, as shown in figure 7. The overall energy is the sum of the energies produced by the two factors.

The first factor is the association module $A(Y, f_1(X), f_2(X))$. The association module is fixed (it has no trainable parameter), and produces a high energy if Y gets very different from either $f_1(X)$, or $f_2(X)$. The energy gets particularly large if $f_1(X)$, and $f_2(X)$ agree

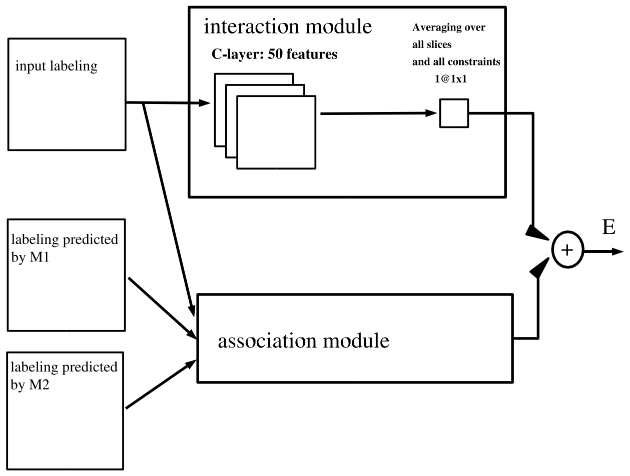


Fig. 7. The architecture the Energy-Based Model. The image marked "input labeling" is the variable to be predicted by the EBM. The first layer of the interaction module is a convolutional layer with 10 feature maps and 5×5 kernels operating on the 5 feature maps from the output label image. The non-linear function for each node is of the form $g(u) = \frac{u^2}{(1+u^2)}$. The second layer simply computes the average value of the first layer.

and Y takes a different value. The module ensures that the cleaned-up label image will not be drastically different from either $f_1(X)$, or $f_2(X)$. The function is encoded in the form of a $5 \times 5 \times 5$ table which contains the energies associated with each possible combination of values of the variables Y, $f_1(X)$, and $f_2(X)$ at any particular pixel location. The output energy of the association module is simply the average of those energies over all pixel locations.

The second factor is the *interaction module* $I(W, Y)$. The interaction module implements the local consistency constraints and is trained from data. The first layer of the interaction module is a convolutional layer with 10 feature maps and 5×5 kernels that operate on the 5 feature maps of Y. The non-linear activation function of the units is of the form $g(u) = \frac{u^2}{(1+u^2)}$. This function and its derivatives are shown in figure 8. The idea behind this activation function is that each unit implements a *linear constraint* on the neighborhood of variables from which it takes inputs [33]. When the input vector is near orthogonal to the weight vector, the output is near zero, indicating that the constraint is satisfied. When the input vector has a non-zero projection on the weight vector, the output is non zero. If the constraint is strongly violated, the output will be near 1.0 (the asymptote). The saturating asymptote ensures that only a "nominal price" will be paid (in terms of energy) for violating a constraint [33]. The total output energy of the interaction module is the average of the outputs of all the 10 feature maps over all positions.

B. Training the EBM

As we mentioned earlier, a suitable loss function must be found whose minimization will "dig holes" in the energy landscape at the location of the desired Y for given $f_1(X)$ and $f_2(X)$, and "build hills" for all other (incorrect) value of Y.

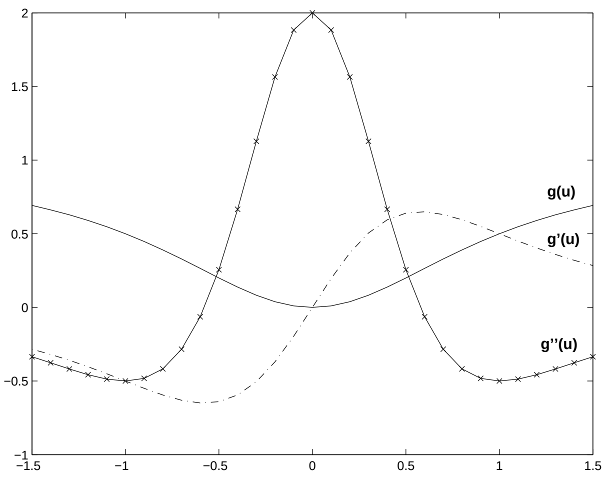


Fig. 8. The activation function $g(u) = \frac{u^2}{1+u^2}$ used in the EBM feature maps.

As reported in [21], there are several suitable loss functions whose minimization will achieve the desired result. For this work, we have used the loss function below. For a given training example (X^i, Y^i), where X^i is an input image and Y^i a human-produced label image, the loss function is:

$$L(W,Y^i,X^i) = E(W,Y^i,f_1(X^i),f_2(X^i)) + c_1 e^{-c_2 \min_{y,y\neq Y^i} E(V} \quad (2)$$

where c_1 and c_2 are user-specified positive constants. The overall loss function is the average of the above function over the training set. The first term is the energy associated with the desired input configurations $(Y^i, f_1(X^i), f_2(X^i))$. Minimizing the loss will make this energy low on average. The second term is a monotonically decreasing function of $\min_{y,y\neq Y^i} E(W,y,f_1(X^i),f_2(X^i))$, which can be seen as the energy of the 'best wrong answer", i.e. the lowest energy associated with a y that is different from the desired answer Y^i. Minimizing the loss will make this energy large. Minimizing this loss function makes the machine approach the desired behavior by making the energy of desired configurations low, and the energy of wrong configurations found by our inference algorithm high.

It takes about 5 hours to complete one iteration of training. Inference is rather fast and it takes less than 1 minute for 10 inference steps (10 updates of each site), which is sufficient for our denoising purpose. Similar loss functions have recently been used in the somewhat different contexts of face detection and pose estimation [28], pose and illumination-invariant generic object recognition [21], and face verification using trainable similarity metrics [4].

C. Results

The training sets and test sets were the same as for the convolutional neural net training. A few results produced by applying the EBM to label image produced by the convolutional network are shown in figure 9. The method does a good job at eliminating isolated points that were erroneously labeled. The nuclei are clearly identifiable in the resulting images. However, the method is a bit overzealous in cleaning up cell wall pixels. Several legitimate cell wall pixels that were correctly identified by the convolutional net were eliminated by the EBM.

It is possible that these deficiencies could be reduced by modifying the architecture, changing the loss function, or improving the training procedure. Because EBM is a relatively new method, many aspects of EBM training are not yet fully understood. More research is needed in this area.

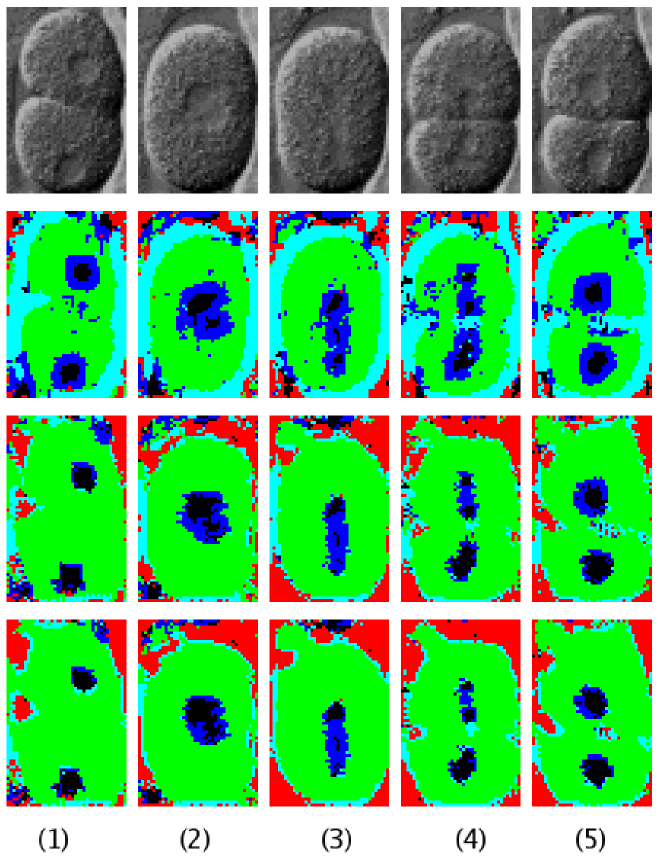


Fig. 9. Results of EBM-based clean-up of label images on 5 testing images. Top line: input image; second line: output of M2 convolutional network; 3rd and 4th line value of Y (cleaned-up image) at two different stages of the energy-minimizing inference process.

IV. Deformable Templates for Global Analysis

The previous sections discuss image analysis techniques to accurately segment images into cellular elements such as cell nucleus, nuclear membrane, etc. Further processing steps are needed to transform this pixel-wise information into the more global characteristics that are relevant to automatic phenotyping. This section discusses various methods we considered for extracting information at such higher levels of abstraction. When experts visually inspect images, they observe the emerging organization of the multi-cellular organism. This information is often conveyed by drawing a sketch of the cell representing the relative shape and position of the cells and of their nuclei. When inspecting a movie, sequences of specific events occuring over multiple frames are identified. Events can be global (for instance, which cells divide and in what order) or very specific (for instance, a visible feature can appear between two specific cells at a specific time during the embryo development).

An possible approach consists in collecting sketches representing various normal or abnormal stages of the organism development and using them as a dictionary of deformable templates that can be aligned and matched with each frame of the movies. Identifying the stage of development comes down to finding the deformable template in the dictionary that best matches the image under consideration. By aliging and fitting the template to the images, we can extract the relative shapes and position of the cells, the orientation of the organism and the identity of each individual cell. Deformable templates also enforce global constraints that are not easily implemented by local analysis. For example, the nucleus must lie roughly in the center of the cell, the cell boundary must be a closed curve, etc.

Deformable template models have been used successfully in several problems, such as medical image processing [14], object matching in video sequences [2], and many other applications. For a survey of active contour methods, see [26].

In the following, we present two different methods for matching label images to deformable templates. The first method uses "sparse" elastic templates that are matched to the label images by minimizing an energy function using an efficient method reminiscent of the Expectation-Minimization algorithm (EM). The second method uses "dense" elastic templates that are matched to the label images using an algorithm reminiscent of Kohonen's Self-Organizing Map algorithm.

A. *Fitting sparse deformable templates with EM*

Matching images to simple deformable templates can be achieved using the Expectation-Maximisation algorithm (EM). [9] have applied spline-based models for hand-written character recognition. They used deformable splines whose control point positions were optimized with the EM. Each spline was seen as the mean of a probabilistic Gaussian model that could generate the "ink" of a character. Similarly, [3] proposed a normalization method for handwritten words that used EM to fit quadratic lines to key points on the trajectory of a pen writing the word.

Experiments were performed with templates defined by a set of key points linked by springs with given rest lengths and stiffnesses. Figure IV-A shows deformable templates for successive stages of the development of the *C. elegans* embryo. Each key point is interpreted as the mean of a Gaussian distribution that can generate pixels of the same label in the label image. A particular template can be viewed in probabilistic terms as a mixture of Gaussian model, where the relative positions of the means of the components Gaussians are dependent on each other. Different springs have been given different spring constants in the models. For example, the springs that link two key points on the cell wall are very stiff ($k = 100$), while the springs that link the nucleus to the cell wall are less stiff ($k = 1$) to allow the nucleus to move around.

We determine the embryo's stage of development by fitting each of those models to an image and finding the model that matches the label image with the lowest energy. The EM fitting algorithm alternates two steps. The *E-step* consists in computing *responsability* coefficients that assign each pixel in the label image to a key point in the deformable template. The *M-step* consists in finding the minimum energy configuration of the spring system given those responsabilities. Since this energy is quadratic in the position of the key points, the solution can be found by solving a linear system. The E and M steps are iterated until convergence. Examples of deformable templates matched to real label images are shown in figure 11.

B. *Fitting dense deformable templates with Colored SOMs*

A second method was tested using templates with "dense" nodes. Such dense templates may finer levels of global information such as the precise position of the cell boundaries. Fitting such models with the EM algorithm is prohibitively expensive, and subject to local minima. Xu and Prince [38] claim that complex contours may be identified more efficiently using dynamic algorithms that do not derive from the optimization of an energy function. Interesting active contour algorithms [1] are derived from Kohonen's Self-Organizing Map method (SOM) [15].

Figure 12 shows preliminary results obtained using Colored SOMs. Each deformable template is specified by assigning labels to the nodes of a regular lattice [37]. The lattice is then aligned with the label images associated with each frame using a variant of Kohonen's SOM algorithm [15]. Each iteration of the alignment algorithm picks a random image pixel and locates the closest lattice node with the same label. This node is then moved towards the location of the image pixel. Neighboring nodes in the lattice are also moved in the same direction with an amplitude that depends on their lattice distance to the node being considered. Step sizes and neighborhood sizes are decreased slightly after each iteration. Results obtained with this method are promising, but preliminary.

V. Discussion and future work

The methods presented in this paper are the first components of a fully automated phenotyping system that can operate on microscopic image sequences of small clumps of cells. A convolutional network labels each pixel as nucleus, nuclear membrane, cytoplasm, cell wall or external medium. An Energy-Based Model post-processes the label images so as to satisfy local consistency constraints. Finally a set of deformable templates are matched to the label image to identify the stage of development of the embryo and to precisely locate the cell nuclei.

The final system will also include a Hidden Markov Model that will be used to classify movies into normal or abnormal development, to identify the type of abnormal development, and to locate the time at which key events take place.

Because labeled data is very scarce, and because the system has not yet been fully integrated, we report results for each of the three modules that are somewhat qualitative. The measurement of meaningful quantitative results will require a

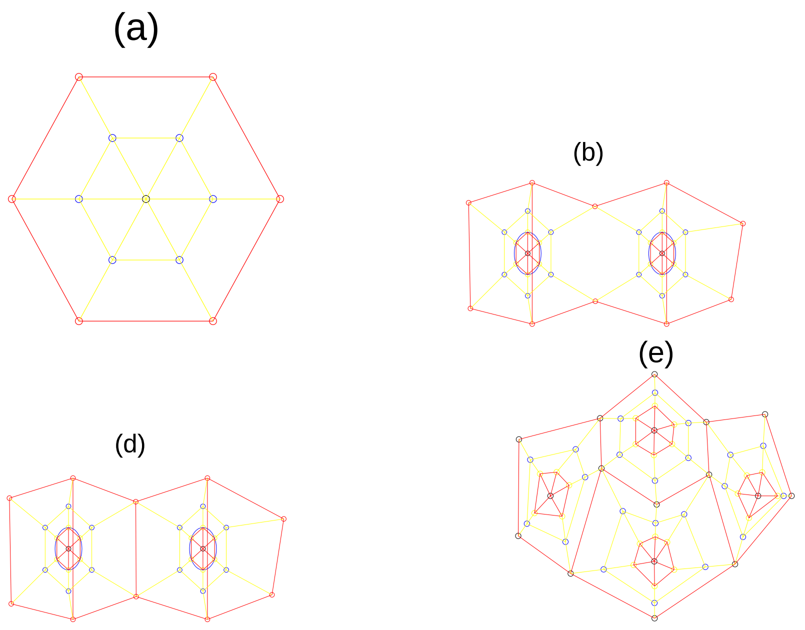


Fig. 10. EM deformable templates for each stage of the *C. elegans* embryo development. (a) Fertilization has just occurred. (b) The maternal pronucleus migrates to the posterior area and a pseudo-cleavage furrow forms. (c) The pronuclei fuse. (d) The cell divides unequally to produce two cells. (e) The two cells further split into four cells.

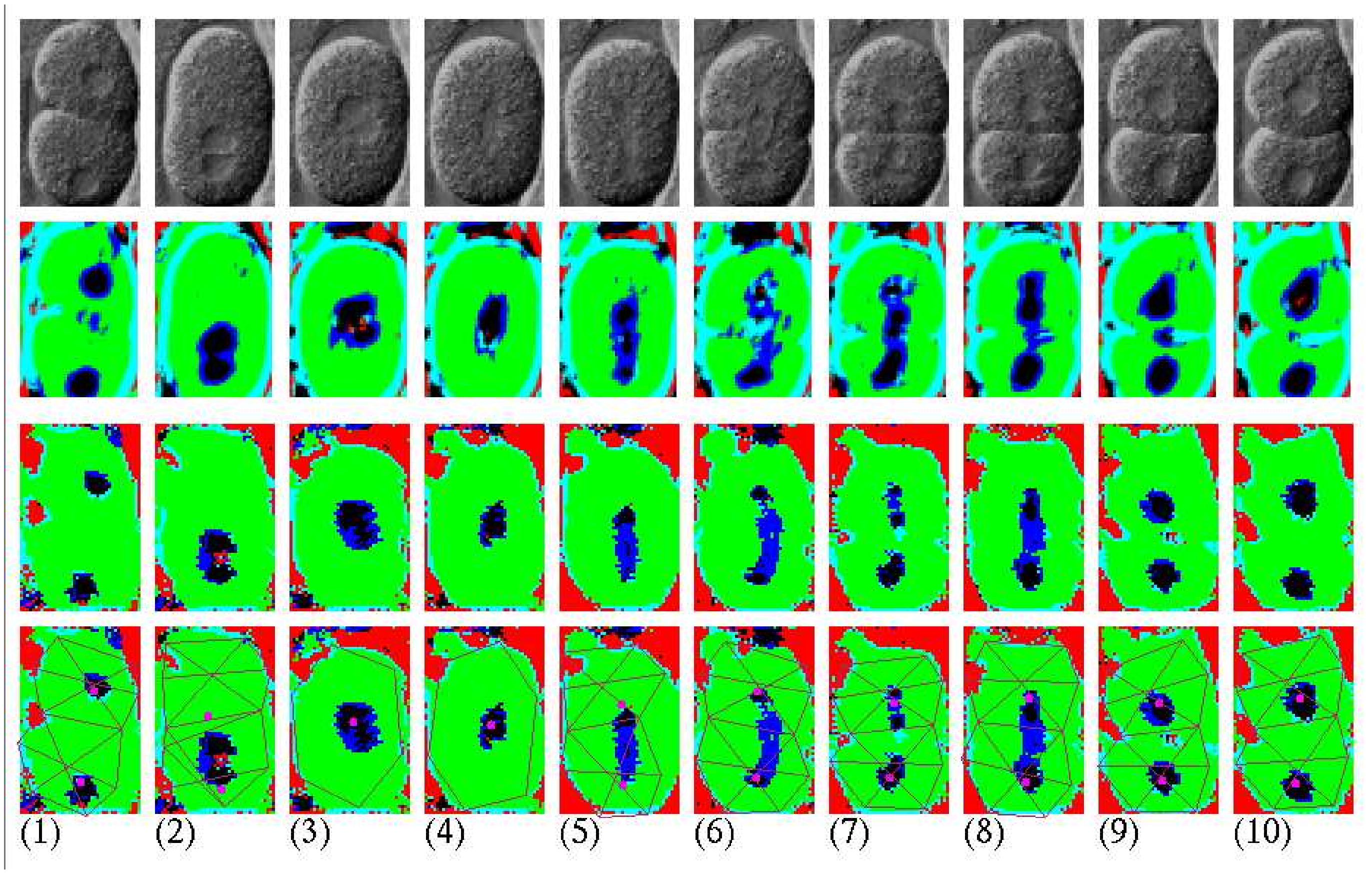


Fig. 11. Matching deformable templates to label images. Top to bottom: input images, M2 predictions, EBM output, EBM output overlaid with deformable template that produced the best fit. Multiple templates are applied to each image. Only the template with smallest energy is shown here. Only the stiff springs of the template are drawn (yellow lines). The pink dots indicate the node of the deformable template that corresponds to the center of the nucleus. We find that the 2-cell templates correctly match image 1, $5-10$, while the 1-cell templates correctly match images $3-4$. However, image 2 has been wrongly matched with a 2-cell template (albeit with a high energy).

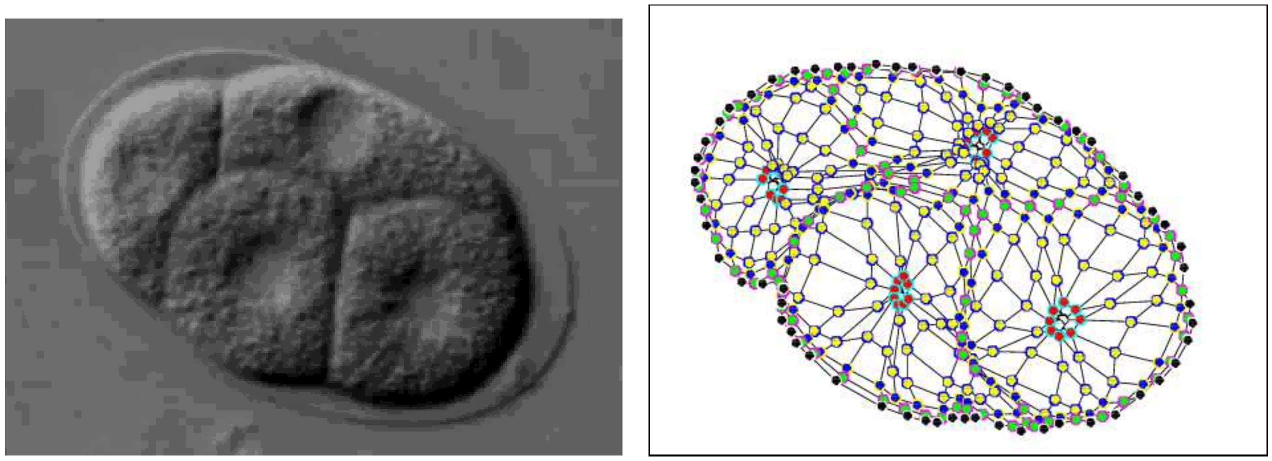


Fig. 12. Matching deformable templates with Colored Self-Organizing Maps. Each deformable template is specified by coloring a regular lattice of nodes. The lattice is then aligned with the cell component labels derived from the image.

considerably larger dataset than is currently available, as well as a fully integrated system.

The main advantage of the approach presented here is that it is fully trainable, and therefore fairly *generic*. Applying the system to a new problem comes down to collecting labeled data and training the system with this new data. A particularly interesting aspect of the approach is that the convolutional network takes care of learning appropriate features and eliminates the need for hand-designed feature sets that may be problem-dependent.

The paper describes one of the first uses of Energy-Based Models for solving a practical task. EBM training methods are very much in their infancy. It is likely that performance improvements will be obtained with better architectures, better loss functions, and better optimization techniques.

A. Future Work

Because the configuration of the embryo changes slowly from frame to frame, it should be possible to improve the reliability of the labeling system by processing several successive frames simultaneously. The system described here could easily be modified so that a window of a few successive frames could be fed to the convolutional network. Similarly, the EBM could take multiple successive frames into account and encode temporal as well as spatial consistency. The main obstacle to this is the small amount of manually-labeled successive frames currently available. Producing more labeled data is one of the current priorities of the present project. Semi-automated software tools are being developed that will accelerate the labeling process.

The goals of a fully automatic phenotyping system for C. Elegans Embryos are (1) to label movies as normal or abnormal, (2) to classify the type of abnormal development, (3) to perform quantitative measurements such as the distance between nuclei, the time of cell division, etc. Building such a system will require modeling the sequential aspect of the data. One technique being pursued involves the use of Hidden Markov Models (HMM). Each known development scenario (normal or abnormal) can be associated with an HMM whose states represent the various stages of embryo development. The emission probability model for each state is a mixture model whose components are the deformable templates. Each deformable template can be seen as a probability density model whose log-likelihood is proportional the fitting energy of the deformable model. Classifying a movie into one of the scenarios simply consist in finding the HMM that maximizes the likelihood of the observed data. This can be performed with one of the standard methods for HMM inference (Viterbi algorithm, or forward algorithm).

B. Conclusion

The emergence of fully-automated phenotyping system will allow very large-scale exploratory experiments in functional genomics. With an automatic phenotyping system, it may become possible to perform systematic experiments where pairs of genes are knocked out, perhaps unveiling new regulatory interactions.

A fully automated system is still several years away, but the methods presented in this paper suggests that it may be within reach. Because the methods presented here are trainable, and not particularly tuned to the particular problem at hand, they may be easily applied to other image-based phenotyping applications.

References

[1] Arnaldo J. Abrantes and Jorge S. Marques. A common framework for snakes and Kohonen networks. In *Proceedings of the 1993 IEEE Conference on Neural Networks for Signal Processing*, pages 251–260, 1993.

[2] A.K.Jain, Y. Zhong, and S. Lakshmanan. Object matching using deformable templates. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 18(3):267–278, Mar 1996.

[3] Y. Bengio and Y. LeCun. word normalization for on-line handwritten word recognition. In IAPR, editor, *Proc. of the International Conference on Pattern Recognition*, volume II, pages 409–413, Jerusalem, October 1994. IEEE.

[4] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verificatio. In *Proc. of Computer Vision and Pattern Recognition Conference*. IEEE Press, 2005.

[5] C. Conrad, H. Erfle, P. Warnat, N. Daigle, T. Lorch, J. Ellenberg, R. Pepperkok, and R. Eils. Automatic identification of subcellular phenotypes on human cell arrays. *Genome Research*, 14:1130–1136, 2004.

[6] The *C. elegans* Sequencing consortium. Genome sequence of the nematode *C. elegans*: a platform for investigating biology. *Science*, 11(282):2012–2018, Dec 1998.

[7] A. Fire, S. Xu, M.K. Montgomery, S.A. Kostas, S.E. Driver, and C.C. Mello. Potent and specific genetic interference by double-stranded rna in *Caenorhabditis elegans*. *Nature*, 391(6669):806–811, Feb 1998.

[8] Christophe Garcia and Manolis Delakis. Convolutional face finder: A neural architecture for fast and robust face detection. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 26(11):1408–1423, Nov 2004.

[9] M. Revow G.E.Hinton, C.K.I.Williams. Adaptive elastic models for hand-printed character recognition. In *Advances in Neural Information Processing Systems (NIPS*1991)*. MIT Press, 1991.

[10] P. Gonczy, C. Eheverri, K. Oegema, A. Coulson, S.J. Jones, R.R. Copley, J. Duperon, M. Brehm, E. Cassin, M. Kirkham, S. Pichler, K. Flohrs, A. Goessen, S. Leidel, A.M. Alleaume, C. Martin, N. Ozlu, P. Bork, and A.A. Hyman. Functional genomic analysis of cell division in *C. elegans* using RNAi of genes on chromosome III. *Nature*, 408(6810):331–336, Nov 2000.

[11] Fu-Jie Huang, Yann LeCun, and Leon Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Proceedings of CVPR'04*. IEEE Press, 2004.

[12] K. Huang and R. F. Murphy. From quantitative microscopy to automated image understanding. *J. Biomed. Optics*, (9):893–912, 2004.

[13] R. A. Hummel and S. W. Zucker. *On the foundations of relaxation labeling processes*, pages 585–605. Morgan Kaufmann Publishers Inc., 1987.

[14] M. Kass, A. Witkin, and D. Terzopoulos. Snake: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1987.

[15] T. Kohonen. *Self-Organization and Associative Memory (2nd edition)*. Springer Verlag, Berlin, New York, 1984.

[16] Sanjiv Kumar and Martial Hebert. Discriminative fields for modeling spatial dependencies in natural images. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

[17] John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. International Conference on Machine Learning (ICML)*, 2001.

[18] Steven. Lawrence, C. Lee Giles, Ah Chung Tsoi, and Andrew Back. Face recognition: A convolutional neural network approach. *IEEE Transactions on Neural Networks*, 8(1):98–113, 1997.

[19] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998.

[20] Y. LeCun, L. Bottou, G. Orr, and K. Muller. Efficient backprop. In G. Orr and Muller K., editors, *Neural Networks: Tricks of the trade*. Springer, 1998.

[21] Yann LeCun and Fu Jie Huang. Loss functions for discriminative training of energy-based models. In *Proc. of the 10-th International Workshop on Artificial Intelligence and Statistics*, 2005.

[22] S. Z. Li. *Markov random field modeling in computer vision*. Springer-Verlag, 1995.

[23] M. K. Markey M. V. Boland and R. F. Murphy. Automated recognition of patterns characteristic of subcellular structures in fluorescence microscopy images. *Cytometry*, (33):366–375, 1998.

[24] D. Martin, C.Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color and texture cues. *Trans. on PAMI*, 26(5):530–549, Jan 2004.

[25] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. Computer Vision*, volume 2, pages 416–423, July 2001.

[26] T. McInerney and D. Terzopoulos. Deformable models in medical image analysis: a survey. *Medical Image Analysis*, 1(2):91–108, Mar 1996.

[27] S. Nowlan and J. Platt. A convolutional neural network hand tracker. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 901–908, San Mateo, CA, 1995. Morgan Kaufmann.

[28] R. Osadchy, M. Miller, and Y. LeCun. Synergistic face detection and pose estimation with energy-based model. In *Advances in Neural Information Processing Systems (NIPS*2004)*. MIT Press, 2005.

[29] M. Pelillo and M. Refice. Learning compatibility coefficients for relaxation labeling processes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 16(9):933–945, 1994.

[30] F. Piano, A.J. Schetter, M. Mangone, L. Stein, and K.J. Kemphues. RNAi analysis of genes expressed in the ovary of *Caenorhabditis elegans*. *Current Biology*, 10(24):1619–22, Dec 2000.

[31] F. Piano, A.J. Schetter, D.G. Morton, K.C. Gunsalus, V. Reinke, S.K. Kim, and K.J. Kemphues. Gene clustering based on rnai phenotypes of ovary-enriched genes in *C. elegans*. *Current Biology*, 12(22):1959–64, Nov 2002.

[32] M. Revow, C. K. I. Williams, and G. E. Hinton. Using generative models for handwritten digit recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(6):592–606, June 1996.

[33] Y. W. Teh, M. Welling, S. Osindero, and Hinton G. E. Energy-based models for sparse overcomplete representations. *Journal of Machine Learning Research*, 4:1235–1260, 2003.

[34] R. Vaillant, C. Monrocq, and Y. LeCun. Original approach for the localization of objects in images. *IEE Proc. on Vision, Image, and Signal Processing*, 141(4):245–250, August 1994.

[35] E. van Munster, L. van Vliet, and J Aten. Reconstruction of optical pathlength distribution from images obtained by a wide-field differential interference contrast microscope. *Journal of Microscopy*, 188(2):149–157, 1997.

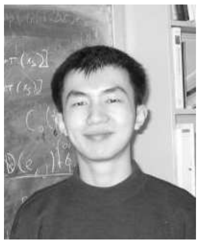
[36] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, pages 511–518, 2001.

[37] J. Vleugels, J. Kok, and M. Overmars. Motion planning using a colored Kohonen network. *International journal of neural systems*, 8:613–628, 1997.

[38] C. Xu and J. Prince. Gradient vector flow: A new external force for snakes. In *Proceedings of Computer Vision and Pattern Recognition (CVPR '97)*, pages 66–71, San Juan, Puerto Rico, June 1997. IEEE.

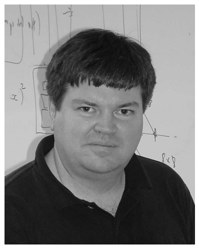
[39] T. Yasuda, H. Bannai, S. Onami, S. Miyano, and S. Kitano. Towards automatic construction of cell-lineage of *C. elegans* from Nomarski DIC microscope images. *Genome Informatics*, 10:144–154, 1999.

[40] P. Zipperlen, A.G. Fraser, R.S. Kamath, M. Martineez-Campos, and J. Ahringer. Roles for 147 embryonic lethal genes on *C. elegans* chromosom I identified by RNA interference and video microscopy. *EMBO Journal*, 20(15):3984–3992, Aug 2001.



Feng Ning Feng Ning is a PhD student of Computer Science with the Courant Institute of Mathematical Sciences at New York University. He received his BS degree in Mathematics from Wuhan University, China in 1998, and a MS degree in Mathematics from Peking University, China in 2001. From 2001 to now, he pursues his PhD study at New York University and now works in Computational and Biological Learning Lab, led by professor Yann LeCun.

Damien Delhomme Damien Delhomme has a Master's degree in Applied Mathematics from the Ecole Centrale de Paris and a Master's degree in Machine Learning from the Ecole Normale de Cachan. He recently started up Naskeo Environnement, a company providing factories with water treatment plants which produce renewable energy.



Yann LeCun Yann LeCun is Professor of Computer Science with the Courant Institute of Mathematical Sciences at New York University. He received a Diplôme d'Ingénieur from Ecole Supérieure d'Ingénieur en Electrotechnique et Electronique, Paris in 1983, and a PhD in Computer Science from Université Pierre et Marie Curie, Paris, in 1987. He has held the positions of Research Associate with the University of Toronto, 1987-1988; Member of Technical Staff at AT&T Bell Laboratories in Holmdel, NJ, 1988-1996; Head of the Image Processing Research Department at AT&T Labs-Research, Middletown, NJ, 1998-2001; and Fellow of the NEC Research Institute, Princeton, NJ, 2002-2003.

Dr. LeCun has published over 100 technical papers on machine learning, pattern recognition, computer vision, neural networks, image compression, digital libraries, and VLSI design. He is one of the creators of the back-propagation learning algorithm, the DjVu image compression technology, and several handwriting recognition systems, one of which is used to automatically process a large proportion of US bank checks. His current interests include computational and biological learning, robotics, and computer vision, particularly object recognition.

He is associate editor of IEEE Trans. PAMI, and the International Journal of Computer Vision. He has served on the board of the IEEE Trans. on Neural Networks, the Machine Learning Journal, and the Journal of Pattern Analysis and Applications. He is general chair of the annual Learning at Snowbird Workshop since 1997. He has served as area chair for IJCNN 89, INNC 90, NIPS 90, 94, and 95, and CVPR 2000 and 2005. He is program co-chair of CVPR 2006, New York City.

Fabio Piano Fabio Piano is Assistant Professor with the Department of Biology and the Center for Comparative Functional Genomics at New York Univeristy. He received his Bachelor degree in Biology in 1988, his Master degree in Biology in 1991 and his PhD in Biology with distinction in 1995, all from New York University. From 1996 to 1999, he worked as a postdoctoral fellow in Cornell University. He was also an adjunct professor of Genetics at University of Florence in 1998. From 1999 to 2001, he was a research associate II at Cornell University. He has been an Assistant Professor at New York University since 2002. He has published more than 20 papers on genetics and related areas.

Léon Bottou Léon Bottou received a Diplôme from Ecole Polytechnique, Paris in 1987, a Magistre en Mathmatiques Fondamentales et Appliques et Informatiques from Ecole Normale Supérieure, Paris in 1988, and a PhD in Computer Science from Université de Paris-Sud in 1991. He joined AT&T Bell Labs from 1991 to 1992 and AT&T Labs from 1995 to 2002. Between 1992 and 1995 he was chairman of Neuristique in Paris, a small company pionneering machine learning for data mining applications. He has been with NEC Labs America in Princeton since 2002. Léon's primary research interest is machine learning. His contributions to this field address theory, algorithms and large scale applications. Léon's secondary research interest is data compression and coding. His best known contribution in this field is the DjVu document compression technology (http://www.djvuzone.org.) Lon published over 60 papers and is serving on the editorial boards of the Journal of Machine Learning Research and of Pattern Recognition Letters.

PLACE
PHOTO
HERE

Paolo Emilio Barbano Paolo Emilio Barbano has received the PhD degree in Mathematics in 1996 from the City University of New York - Graduate Center, New York, NY. He has been an Assistant Professor of Mathamatics at Yale University from 1998 to 2001. He is currently affiliated with the Yale Department Mathematics. Dr. Barbano's main research interests are Mathematical Signal Processing for Radar and Communications as well as Learning Techniques applied to Digital Signal Processing.