A Deep Framework for Bacterial Image Segmentation and Classification

Dong Nie
Department of Computer Science
University of North Carolina at Chapel Hill
Chapel Hill, NC 27599
dongnie@cs.unc.edu

Elizabeth A. Shank
Department of Biology
University of North Carolina at Chapel Hill
Chapel Hill, NC 27599
eshank@email.unc.edu

Vladimir Jojic
Department of Computer Science
The University of North Carolina at Chapel Hill
Chapel Hill, NC 27514
vjojic@cs.unc.edu

ABSTRACT

Bacterial image segmentation and classification is an important problem because bacterial appearance can vary dramatically based on environmental conditions. Further, newly isolated species may exhibit phenotypes previously unseen. Conventionally, biologists identify bacteria using colony morphology, biochemical properties, or molecular phylogenetic approaches. However, these phylogenetic classification approaches do not provide predictive information about the colony morphology that is expected to result from the growth of these bacteria on agar, or how these morphological phenotypes might vary in the presence of other bacterial species. In this paper, we propose a framework to automatically identify and classify regions of bacterial colony images and correspond them across different images from different contexts. Importantly, this approach does not require prior knowledge of species' appearances. Rather, our method assumes that images contain one or more bacteria from a pool of bacteria, and *learns* morphological features relevant to distinguishing between bacteria in this pool.

Our method first segments the image into regions covering the bacterial colonies, agar, plate, and various border artefacts. To achieve this, we use an unsupervised deep learning technique, Convolutional Deep Belief Network (CDBN). This technique provides a deep representation of small image patches. Using this high-level representation instead of raw pixel intensities, we train a support vector machine (SVM). The trained SVM accurately classifies foreground and background patches.

Once the foreground patches are identified, we train a supervised deep learning method, Convolutional Neural Network (CNN), that predicts which bacterial colonies from the pool occur in a query image. Experimental results demonstrate that our method outperforms other classical methods on segmentation and classification.

Categories and Subject Descriptors

J.3 [**Life And Medical Sciences**]: Biology and genetics; I.4.7 [**Image Process And Computer Vision**]: Feature Measurement—*Feature representation*

BCB'15, September 9–12, 2015, Atlanta, GA, USA.

http://dx.doi.org/10.1145/2808719.2808751.

General Terms

Design,Experimentation

Keywords

deep learning, bacterial colonies, segmentation, classification

1. INTRODUCTION

Identification of bacteria is an important problem in biology. It is essential for disease diagnosis, treatment of infection, and traceback of disease outbreaks associated with microbial infections. Biologists have traditionally identified bacteria using phylogenetic molecular methods (i.e. sequencing the 16S rRNA gene). However, in many cases knowing the phylogenetic identity of bacteria is not sufficient or necessary. Instead, being able to describe and track the morphological and visual characteristics of bacterial cells and their resultant colonies is the priority. Machine learning methods are ideally suited for training algorithms to discriminate bacteria from appearance alone. [9] uses artificial neural networks for bacterial identification. [16] introduces an automatic approach to identify and classify the bacterial growth phases of bacillus-shaped cells in digital microscopic cell images using a number of handcrafted geometric features and a variant of Naive Bayes method. [33] proposes a statistical image analysis method for automatic identification of bacterial types. However, the previous research mainly uses low-level features, such as sets of shape features [14, 15] and related geometric shape features [25]. Even though these features can be calculated efficiently and lead to classifiers with good performance in some simple situations, they don't perform well in colonies exhibiting deformable and changeable morphologies.

This is in part because the morphological features of bacterial colonies change over time, leading to what we refer to here as multiple 'growth phases'. In addition, the interactions of these bacterial colonies with other microbes can lead to further changes in morphology. Analysis of bacterial colony morphology changes as a result of microbe-microbe interactions has became a popular research topic [5] and is important for identifying which bacterial pairs may be interacting via chemical or physical signals that may alter the composition and function of microbial communities. Traditionally, biologists merely visually observed the effects of these interactions. However, making such observations is a subjective and time-consuming process. Thus an objective automatic identification method would be very helpful and facilitate the identification of those bacterial interactions that may be occurring through important cell-cell signaling processes.

In this paper, we propose a method for the automatic identification and classification of bacterial colonies in digital images using high-level features that capture morphological motifs. In contrast

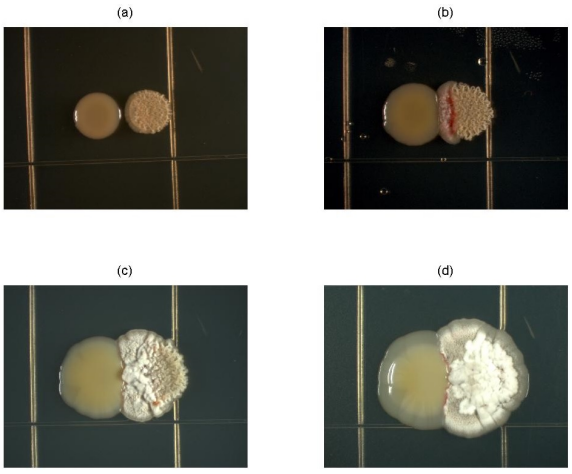

Figure 1: Examples of images taken on different days: a) 2^{nd} day, b) 4^{th} day, c) 6^{th} day, d) 8^{th} day

to previous work, we do not hand-design the features. Rather, we rely on deep learning [22] to discover high-level features that characterize the different bacterial strains. Specifically, in the first step, we apply an unsupervised deep learning approach (convolutional deep belief network or CDBN) to locate the portions of the image that capture one or more bacterial colonies. Then we utilize a supervised deep learning method (convolutional neural network) to distinguish between bacterial colonies based on appearance alone. Real-data experiments were used to test our proposed deep framework. The scientific contribution of our work is: 1) this is the first application of the deep learning framework to learn high-level features of bacterial colony appearance, 2) we show that using these features leads to superior segmentation and identification methods 3) the high-level features provide a basis for the analysis of colony morphology changes induced by cross-species interactions.

This paper is organized as follows. Section 2 introduces the materials and basic image processing. Section 3 gives an overview of the main approaches explored in this work. Details of the methods are presented in Section 4 and Section 5. Experiments and results are reported in Section 6. Section 7 concludes our work.

2. DATA PREPARATION

We cultured bacteria on two different media (ISP2 and Oatmeal agar). Each plate contained two colonies selected from a set of 17 strains (11 Streptomycete spp., 1 Micromonospora spp., 1 Amycolatopsis spp., 1 Dermacoccus spp., 1 Actinoplanes spp., 1 Kutzneria spp., 1 Arthrobacter spp.). 1 uL of cell resuspensions at OD600 = 0.5 were put onto agar, at a spacing of 0.5 cm. Mono-culture controls were also spotted. Plates were incubated at 30 [put in degree symbol here]C for 8 days. The plates were imaged at 2, 4, 6, and 8 days after spotting. This yielded a total of 862 images of growing colonies. Fig. 1 show examples of the bacterial colony images at four different growth phases.

In order to classify bacteria we first must detect where they occur in the image. We consider the portions of the image that contain bacteria to be "foreground", and the portions of the image that contain media or plate to be "background". The first task is to segment the image into foreground and background. Training a method that accomplishes this requires labeled examples of foreground and background patches. We manually labeled foreground and background in a small subset of images. The choice of examples to label has direct bearing on the performance of the segmentation algorithm. For example, not providing examples of artefacts such as air bubbles in agar would lead to segmentation that might classify them as foreground. Hence, we chose to label images so that each bacterial colony type and each growth phase was covered, but not necessarily in all contexts.

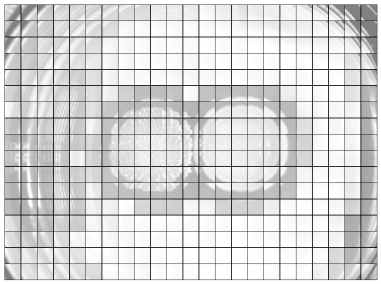

Figure 2: Patch extraction example: patches are extracted from a whole image in gray-scale, and the patches are whitened to remove strong local correlations in intensities.

2.1 Patch Extraction

Since, each plate is spotted with two bacterial colonies, we can not assume that there is only one species in each image. Features computed on the image level would conflate statistics of appearance of the two species and background. We need to localize the features that we compute to particular regions of the image. Hence, we collect patch-level information about the image.

When working with patches, there is a tradeoff between efficiency and accuracy. If patches are large, the number of patches will be smaller, the objects in each patch will be complex, and their processing will be computationally costly. On the other hand, if patches are too small, they may not capture enough information, and their number will be larger. In order to achieve middle ground of having small patches with non-trivial objects, we resized the images to be 1/4 of their original size. Given that the position of the bacterial colonies may shift from an image to image, we collected the candidate patches with overlap.

Our images exhibited strong correlations between nearby pixels, for example agar pixels are quite similar and highly correlated in their intensities. This is a common issue with natural images and it is addressed by the process of whitening, see for example [20]. Fig. 2 is an example of extracted patches from a day 2 image. (Note, to avoid clutter, we do not show all of the overlapping patches).

3. METHODS OVERVIEW

In Fig. 3, we show a diagram of our framework for bacterial image segmentation. Given a series of images, we extract patches with overlap. An unsupervised model, Convolutional Deep Belief Network (CDBN), is trained to obtain the features for the preprocessed patches. A patch-level SVM model is trained to predict background/foreground for the patches represented by features obtained by the CDBN. We refer to the features that CDBN produces for a patch as segmentation-representation.

Once the predicted foreground patches are identified, we can proceed with predicting bacterial species in those patches. Fig. 4 shows a diagram of our framework for this task. Given species labeling for foreground patches, we train CNN to extract features predictive of labels. These features are used in a patch-level identification. We call these features classification-representation.

Given a query image, patches are mapped into the segmentation-specific representation and labeled as foreground or background.

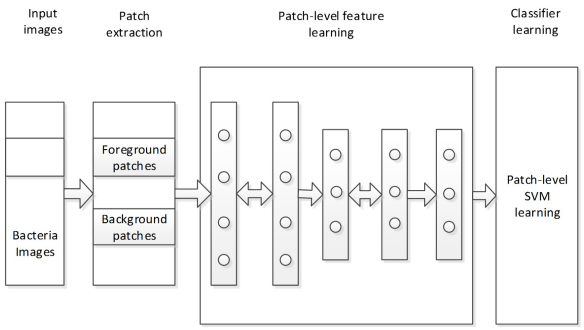

Figure 3: Training pipeline for background/foreground segmentation of bacterial images. We assume that bacterial images in the training set are labeled for foreground and background. The CDBN performs inference on each of the patches to produce values in the higher layers of the network. The values that the neural units assume in the higher layers are deemed segmentation-representation, and fed as features of the patches in training an SVM.

The patches labeled as foreground are then mapped into classification-representation and each is assigned species label. Each patch is assumed to cover a single species. While this assumption may get violated by patches spanning the boundary, their exclusion from the training set can ensure that the classifiers are trained only on unambiguous patches. A vote across all patches provides a ranked list of likely species present in the image.

4. BACTERIAL IMAGE SEGMENTATION

Shown in Fig. 3, CDBN is used to extract features for the bacterial images. Specifically, we designed a convolutional restricted Boltzmann machine (CRBM). First and third layers are convolutional. Second and fourth are max pooling layers. The pooling layers aggregate information from nearby units. In our case outputs from a 2×2 grid of units are passed through a maximum function, thus making our method less sensitive to small shifts. Final layer is a softmax layer which aggregates features into a prediction of the bacterial species. Next, we introduce some background on deep learning models; we review the CRBM and CDBN [24].

4.1 Convolutional RBM and DBN

The CRBM is an extension of the restricted Boltzmann machine (RBM)[31]. An RBM is a two-layer undirected graphical model with visible and hidden units or variables in each layer (corresponding to input data, such as image pixels). Connections exist between visible and hidden layers, but not within hidden nor within visible layers. Data, such as images, can have objects occurring in different, shifted location and still carry the same information. Hence, object detectors need to be *shift invariant*. Convolutional neural networks capture this intuition by applying the same filters throughout different locations in the image, see for example references [28, 24]. Specifically, rather than fully connecting the hidden layer and visible layer, the weights between the hidden units and the visible units are local. For example, a filter of size 10×10 would connect each contiguous patch of this size in the visible layer to a single hidden layer unit. Multiple such filters are typically used in a single network, each responsible for a different object type.

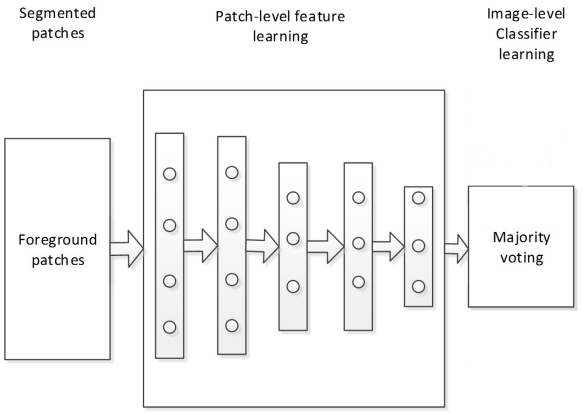

Figure 4: Training pipeline for bacteria type prediction from bacterial images. Only foreground patches are considered in prediction of species appearing in the images. The convolutional neural net trained to predict species in a single patch. Note that a patch is assumed to cover only one species.

We define the joint distribution of a CRBM with real-valued visible units as Eq. 1:

$$P(v;\theta) = \frac{1}{Z}\sum_{h}\exp(-E(v,h)) \tag{1}$$

where the energy function is given by Eq. 2

$$\begin{aligned} E(v,h) = \frac{1}{2}\sum_{i=1}^{N_v} {v_i}^2 - \sum_{k=1}^{K}\sum_{j=1}^{N_H}\sum_{r=1}^{N_w} h_j^k W_r^k v_{j+r-1} \\ - \sum_{k=1}^{K} b_k \sum_{j=1}^{N_H} h_j^k - c\sum_{i=1}^{N_v} v_i \end{aligned} \tag{2}$$

where K is the number of filters, N_v is the number of visible units, N_w is the square of filter size, N_H is the number of hidden units in each group. We note that the *connections* take an explicit form in the above energy function as products between hidden and visible layer units.

Following the energy function, the conditional probability can be computed by Eq. 3 and 4:

$$p\left(h_j^k|v\right) = \text{Sigmoid}(b_k + \sum_{r=1}^{N_w} W_r^k v_{j+r-1}) \tag{3}$$

and

$$p\left(v_i|h\right) = \text{Normal}(c + \sum_{k=1}^{K}\sum_{r=1}^{N_w} W_r^k h_{r-i+1}^k, 1) \tag{4}$$

where $\text{Sigmoid}(\cdot) = \frac{\exp(\cdot)}{1+\exp(\cdot)}$ and $\text{Normal}\,(\cdot)$ is a normal distribution density.

[24] shows that sparsity can greatly improve the performance of CRBM in unsupervised learning tasks. Thus, we integrate sparsity in the hidden layers. The log likelihood for overall model is given by Eq. 5

$$L(v;\theta) = -\log P(v;\theta) - \lambda\left|p - \sum_{k=1}^{K}\sum_{j=1}^{N_H} E\left[h_j^{(k)}|v\right]\right|^2 \tag{5}$$

where p is the sparsity target.

Exact computation of Eq. 5 is intractable. One tractable approximation of this objective is contrastive divergence (CD, see [11]).

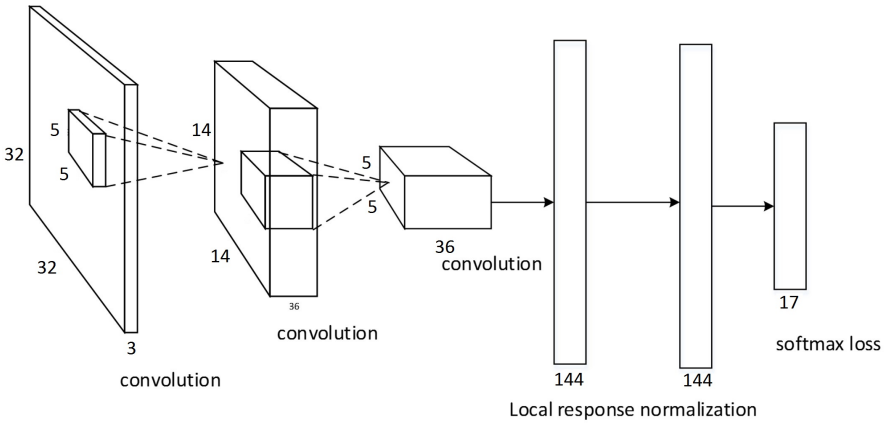

Figure 5: CNN architecture for bacteria patches prediction

Its gradients are fairly inexpensive to compute and its minimization has been empirically shown to provide approximate optima for Eq. 5. More details of the overall training procedure can be seen in [32, 28].

4.2 Layerwise training for multi-layer CRBMs

The above presents an approach to training a single CRBM. Our diagram shows a stack of two CRBMs. Fortunately, [12] proposed a greedy layer-wise algorithm that works for stacks of RBMs. In this procedure, after training a CRBM, we can freeze the first layer's parameters and use it to compute the posterior of the hidden units given the input data. The hidden units' "activation" can be used as input to further train the next layer CRBM. An additional step of pooling activations can be added prior to training next layers.

In our model we use three CRBMs: two convolutional linear RBMs, and one Bernouli RBM. A Bernoulli RBM uses Eq 6 in place of Eq 4, but the training algorithm is the same.

Once all the layers are trained, the CDBN model is built.

$$p\left(v_i|h\right) = Sigmoid(c + \sum_{k=1}^{K}\sum_{r=1}^{N_w} W_r^k h_{r-i+1}^k) \qquad (6)$$

We note that typical applications of CDBN models trained by unsupervised learning is to convert the raw data into a new representation. This representation is obtained by concatenating states of all hidden units for a particular input data.

4.3 Architecture and hyper-parameters

[2] shows that some functions cannot be efficiently represented (in terms of the number of tunable elements) by shallow architectures. The deep architectures have much better representation ability.

However, one of the challenges when using a deep learning framework is selection for architecture and optimal model hyper-parameters. For each layer of neural networks, we must decide the size of the filters, number of filters, max-pooling region size, and sparsity of the hidden units. [30] shows correlation between performance with random filters and learned filters for a given architecture, and suggested using search over architectures with random filters as a proxy for selecting the optimal architecture to use with learned weights. After evaluating the correlation between random weight and learned weight performance for a one-layer network with eight different architectures with adapted optimal hyper-parameters by performing a coarse search over the possible values.

4.4 Patch-level SVM learning

Once we train CDBN, we are in position to map raw patches into a high-level representation. This is accomplished by feeding each patch to the CDBN model and performing Gibbs sampling [7] on latent layers. We perform inference of latent layer variables on each patch for which we have foreground or background label. These inferred values are new features of the patch. Based on these patch features and labels, we train a patch-level SVM binary classifier.

5. BACTERIAL TYPE PREDICTION

We assume that only foreground patches are relevant in the task of species prediction. While it is possible that different strains can have different effect on the medium surrounding the colonies, we do not consider appearance of agar in this work. CNNs are utilized to extract features from these foreground patches. As each image contains two bacteria species, each image is labeled with two independent labels, indicating species present. Further each image patch is labeled with a particular species that it captures. Hence, local prediction task focuses on predicting label of a patch, while the image level prediction task needs to predict one or more species present in the image. For this purpose, we aggregate patch-level species predictions into votes for particular species and rank the species based on the number of patches that voted for them.

5.1 Convolutional neural network

Deep learning models can learn a hierarchy of features – high-level features building on low-level ones. The CNNs [23, 21, 3] are a type of deep models, in which trainable filters and local neighborhood pooling operations are applied in an alternating sequence starting with the raw input images. This results in a hierarchy of increasingly complex features. One property of CNNs is that they can capture highly nonlinear mappings between inputs and outputs [23]. When trained with appropriate regularization, CNNs can achieve superior performance on visual object recognition and image classification tasks [23, 21]. CNNs have been used in other applications. In [17, 18, 34, 10], CNNs were applied to restore and segment the volumetric electron microscopy images; [4] applied deep CNNs to detect mitosis in breast histology images by using pixel classifiers based on patches. In our approach, CNN is utilized to assign the identity of bacterial species to foreground patches.

The choice of activation function has a strong impact on both computational cost of training and prediction performance of CNN models. Common choices are hyperbolic tangent function or sigmoid function. However, models with these nonlinearity func-

tions can be slow to train. As a result, we use Rectified linear unit ($f(x) = \max(0, x)$) [27] in our model.

Since our ultimate goal is to predict labels of patches, we use softmax loss to optimize our CNN model. Softmax regression is an expansion of logistic regression. Softmax loss function is given by Eq. 7

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=1}^{k} 1\{y=j\}\log\frac{\exp(\theta_c^T x)}{\sum_{l=1}^{k}\exp(\theta_l^T x)}\right] \quad (7)$$

where m is the number of examples, and $1\{\cdot\}$ is an indicator function, and x is input x, target variable is $y \in \{1, 2, ..., k\}$, with k denoting the number of distinct classes.

5.2 Deep CNN architectures

In the following, we provide details of our CNN architectures. We use an input patch size of 32×32. These patches cover regions of size 128×128 in the original image, but are substantially smaller due to resizing of the image. The detailed architecture is shown in Fig 5. We trained a CNN model for each growth phase. Hence, we had four input feature maps corresponding to growth phase 1 (the 2nd day), growth phase 2 (the 4th day), and growth phase 3 (the 6the day) and growth phase 4 (the 8th day). The CNN model consisted of three convolutional layers and one fully connected layer. This network also applies local response normalization [21], and softmax layers.

The first convolutional layer contains 36 feature maps. Each of the feature maps is connected to all of the input feature maps through filters of size 5×5. We use a stride size of one pixel. This yields outputs of size 28×28. These outputs are passed through a pooling layer of size 2×2, which produces feature maps of size 14×14. Since there are 36 filters, we obtain one such feature map for each filter. Note that each filter has a set of weights of size $5 \times 5 \times 3$; the last dimensions spans the red, green, and blue intensities.

Thus output of the first set of layers involving convolution and pooling yields values organized into an three dimensional array of size $14 \times 14 \times 36$. This array is fed through another set of convolutions and pooling leading to feature maps of size $5 \times 5 \times 36$. We note that each of the filters in this layer has a set of weight of size $5 \times 5 \times 36$, where the last dimension spans outputs of the 36 filters in the previous layer.

Finally, 144 filters of size 5×5 are applied to the outputs of the second layer, and these in turn are connected through local response normalization to the fourth layer. In encourages competition between features at the same spatial location across different feature maps. The fifth layer consists of the softmax units, aimed at predicting one out of possible 17 labels – bacterial species.

Our network minimizes the softmax loss between the predicted label and ground truth label. In addition, we use dropout [13] to learn more robust features and reduce overfitting. This technique sets the output of each neuron to zero with probability 0.5. The dropout is applied before the fully-connected layers in the CNN architecture of 5. In total, the number of trainable parameters for this architecture is 188,264.

5.3 Majority vote for image-level classification

The CNN model can predict the labels for patches, but the per-patch accuracy varies based on the content of a patch. However, when this information is aggregated across the whole set of foreground patches, the uncertainty diminishes. Hence we collect aggregate outputs from CNN for each patch as votes for predicted species. Thus it is possible that a few patches generate votes for an unlikely species, but overall these unlikely species get out-voted by species whose signatures consistently occur across a number of patches. Further, these votes provide a ranked list. The two species with most votes are predicted to be in the image.

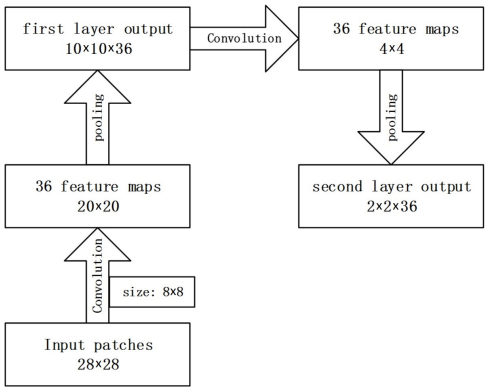

Figure 7: Architecture of convolutional deep-belief network for bacteria images

6. EXPERIMENTS

The whole pipeline of training and prediction in our experiments is captured in Fig. 6.

The following sections will describe details of segmentation and classification evaluation.

6.1 Experiments on segmentation

We chose patches of size w to be 28. According to Section 4.3, we design the CDBN architecture in Fig. 7. To make the CDBN training process converge quickly, we utilize the Gaussian mixture model to initialize training of the first layer CRBM [32, 29].

After the CDBN model is trained, we built our SVM model based on the loosely labeled image dataset. We use the trained CDBN model to extract features of the patches from the foreground/background labeled images, and each patch is represented by joining the first and second layer features. Together with the labels for the patches, we take advantage of liblinear [6] to build a SVM model for binary classification, and cross validation is used to select the optimal parameters for our model.

6.2 Results on segmentation

To compare our proposed methods, Scale-invariant feature transform (SIFT) [26] is used to extract features from the patches. SVM is utilized for classification of the patches based on the SIFT features.

The evaluation for image segmentation can be judged by predictive accuracy. However, the foreground region of an image is relatively small, and the misclassification for foreground patches has significant impact on accuracy. Hence, we also consider misclassification rate, sensitivity, and specificity. Let TP, TN, FP, and FN denote True Positive, True Negative, False Positive, and False Negative, respectively. We present the performances of the competing methods in Table 1.

1. ACCuracy (ACC)= (TP + TN) / (TP + TN + FP + FN)
2. SENsitivity (SEN) = TP / (TP + FN)
3. SPECificity (SPEC)= TN / (TN + FP)
4. Balanced ACcuracy (BAC)= (SEN + SPEC) / 2
5. Positive Predictive Value (PPV)= TP / (TP + FP)

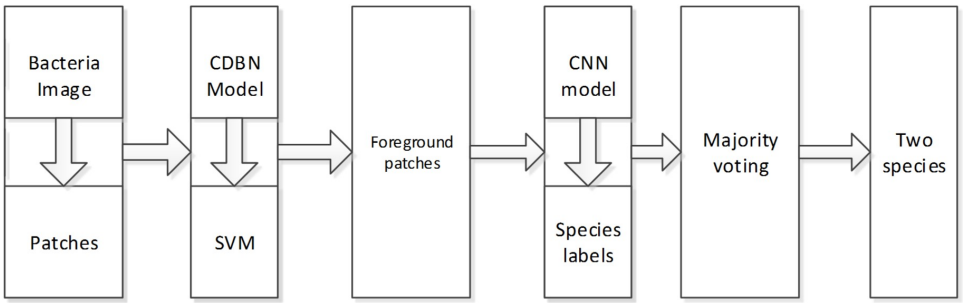

Figure 6: The flow chart for our bacteria image classification system

6. Negative Predictive Value (NPV) = TN / (TN + FN)

The proposed method showed the mean accuracy of 97.14%, beating the accuracy of 88.23%, achieved by the method using SIFT features.

Regarding sensitivity and specificity, the higher the sensitivity, the lower the chance of misclassifying the foreground patches; also the higher the specificity, the lower the chance of misclassifying the background patches. The proposed method had a 10.29% sensitivity than that of SIFT-based method, and a 9.36% higher specificity. In general, the proposed method showed higher sensitivity and specificity in all three classification problems. Hence, from the point of view of segmentation, the proposed method is less likely to misclassify foreground patches as background and vice versa compared to SIFT-based method.

To visualize the segmentation result directly, we recover the predicted patches to a segmentation image, and Fig. 8 is an example of the segmentation for an image of a single bacterial colony.

Why does the proposed method outperform the classical image feature extraction method to such a large degree? In brief, CRBMs take spatial relationships into considerations and learn the high-level features. SIFT features leverage very coarse spatial relationships, whereas CRBMs build a hierarchical representation that adapts to the data's spatial dependencies. As a consequence, in Fig. 8, our proposed method segments the bacteria areas smoothly, while the results from SIFT-based method varies from location to location.

6.3 Experiments on bacterial type prediction

The experiments mainly focus on training CNN models with different growth phase data. The CNN architecture is presented Fig. 5, which is decided after testing eight different network structures following the same method described in Section 4.3. The training process has taken advantage of Caffe [19], which is a commonly used high-speed deep-learning framework.

6.4 Results on bacterial type prediction

In these experiments, we focus on evaluating our CNN architectures for classifying the seventeen types of bacteria species at 4 different growth phases. We formulated the prediction of bacterial species classes as a seventeen-class classification task.

Table 1: A summary of the performances of two methods. The boldface denotes the best performance on each metric for each classification task.

	CRBM features	SIFT features
ACC(%)	$\mathbf{97.14 \pm 2.25}$	88.10 ± 2.52
SEN(%)	**95.16**	84.87
SPEC(%)	**97.59**	88.23
BAC(%)	**96.37**	86.55
PPV(%)	**89.90**	63.11
NPV(%)	**98.90**	96.31

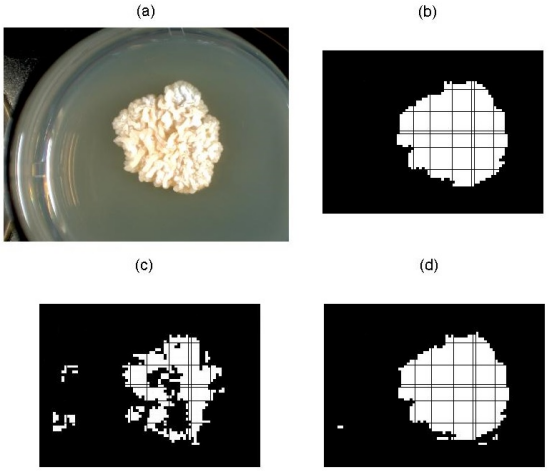

Figure 8: Example of segmentation for background/foreground: (a) is the original bacteria image, (b) is the ground truth, (c) shows the segmentation produced by SIFT method and (d) shows segmentation produced by our proposed method

For comparison purposes, we extract features with SIFT [26], and we implement two other commonly used classification methods, namely the SVM [6] and K-nearest neighbor (KNN) [1]. The performance of SVM was generated by tuning the regularization parameters using cross validation. KNN is a nonparametric algorithm which can be implemented easily, and the K value is set to 5 using cross validation.

The following metrics [8] are used for the evaluation of the bacteria species prediction:

$$
\begin{aligned}
ACCuracy(H,D) &= \frac{1}{|D|}\sum_{i=1}^{|D|}\frac{|Y_i \cap Z_i|}{|Y_i \cup Z_i|} \\
PRECision(H,D) &= \frac{1}{|D|}\sum_{i=1}^{|D|}\frac{|Y_i \cap Z_i|}{|Z_i|} \\
RECall(H,D) &= \frac{1}{|D|}\sum_{i=1}^{|D|}\frac{|Y_i \cap Z_i|}{|Y_i|}
\end{aligned}
\tag{8}
$$

where D is the evaluation data set(bacteria images), consisting of $|D|$ multi-label examples (x_i, Y_i),$i = 1..|D|$,$Y_i \subset L$. Let H be the classifier,$Z_i = H(x_i)$.

The performances of each method is listed in Tab. 2.

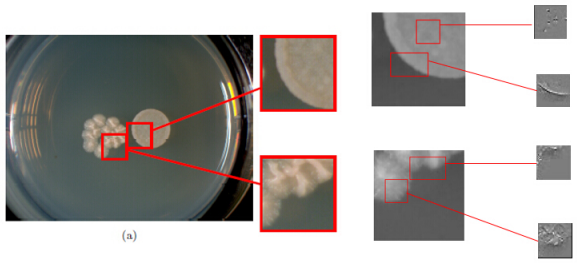

Figure 9: Features learned by CNN model

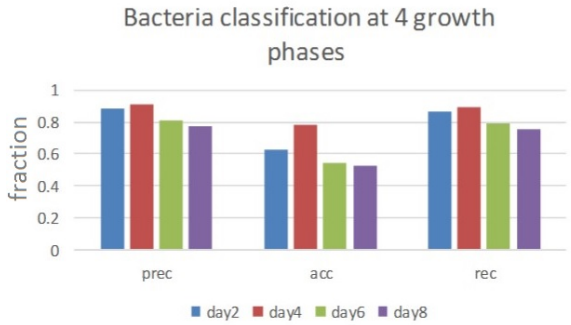

Figure 10: Comparison of classification results for different growth phase data with three metrics: day2, day4, day6, day8 represents the four growth phases

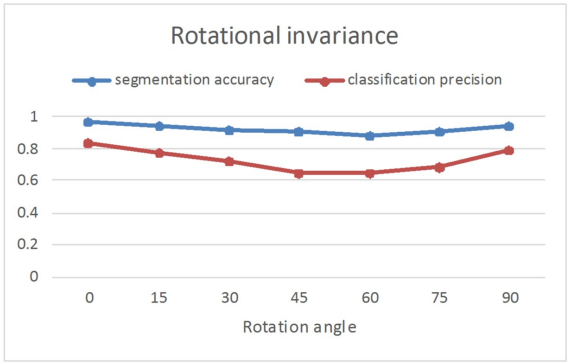

Figure 11: Rotational invariance: images were rotated by 15, 30, 45, 60, 75 and 90 degrees and we computed error on the two tasks, segmentation and classification. Dips in performance can be seen suggesting that parts of representations our method learn had a particular orientation. This suggests that in more complex experiments with multiple interacting colonies capturing axis of interaction may have impact on morphology and consequently species prediction.

CNN method outperforms the two other classification methods. The reason is easy to understand: CNN extracts high-level features, while SIFT only extracts low-level cues [21]. Fig. 9 shows the learned features by CNN model. This figure shows clearly that features learned by CNN model are high-level instead of just simple cues or gradient, the two edges in the feature space are close to the ones in the image space, and the bottom feature patch learns the detailed essence of the original part. Thus, it is easy to understand why deep-learning methods perform better than traditional image feature extraction methods.

We also compare the classification results for bacterial colony images at different growth phases. The comparison is shown in Fig. 10.

Precision reflects exactness of classifiers, *recall* reflects completeness, and *accuracy* reflects how close the predicted label sets are to the ground truth set. Precision for classifying all four phases of the growth data ranges from 77.49% to 91.32%; recall ranges from 75.11% to 89.66%; and accuracy ranges from 52.5% to 78.47%. It is curious that the bacterial species identification at "day2" and "day4" growth phases achieve a higher predictive performance than the later two growth phases, and that the "day4" image set provides the best performance. In summary, the prediction for the interacting bacterial colonies non-interaction bacteria is easier than interacting bacterial colonies, and the more patches that cover the interacting region in an image, the worse the prediction precision is. This phenomenon can be attributed to the interaction area of the image having an unusual appearance, making accurate prediction difficult. Therefore, understanding the impact of bacterial interactions on individual colony appearance is important.

6.5 Invariance analysis

Orientation of bacterial colonies can vary across images. Hence, we asked whether our representations are dependent on a particular orientation of the colonies, or whether it possesses rotational invariance without additional data manipulation.

To test the rotational invariance, we rotated our images by different angles in a clock-wise direction, and then ran the whole pipeline (segmentation and species prediction), for each rotation angle. We then evaluated the performance for segmentation and classification. The result is shown in Fig. 11.

Table 2: A summary of the performance of the three methods. The boldface denotes the best performance in each metric for each task.

	ACC(%)	PREC(%)	REC(%)
CNN	**0.6210**	**0.8376**	**0.8216**
SIFT+KNN	0.4967	0.7124	0.6984
SIFT+SVM	0.5400	0.7725	0.7671

For image segmentation and bacterial-type classification, the rotation of images weakens the performances to some extent. Compared to the original images, the performances of rotated images become worse, and is worst at 45 degrees, but increased after rotation by more than 60 degrees. The performances decreased at most by 9.17% for segmentation and 22.54% for bacterial type classification, however the performances are still competitive compared to SIFT features. This illustrates rotational invariance of the deep-learning features. It also highlights the existence of parts of the image representation that are dependent on the axis of interaction.

7. CONCLUSION AND FUTURE WORK

In this study, we aimed to segment and classify bacterial images across different growth phases and environmental contexts. Image segmentation was achieved by employing CDBN to retrieve sparse representations for patches, and classification was achieved by using a CNN model with multiple intermediate layers. CNNs use the segmented patches from the CDBN model as input, and predict the bacterial species on per-patch level, which are in turn aggregated through a voting scheme to predict the likely species present in an image. We compared the performance of our approach with multiple commonly used image segmentation and classification methods. Results show that our proposed model significantly outperforms the

other methods on bacterial colony image segmentation and classification. Overall, our experiments demonstrate that the proposed deep-learning framework can produce quantitative computational models of bacterial appearance yielding accurate bacterial appearance segmentation and classification tasks. Our model is less efficient at classifying bacterial colony images after they closely physically interact because the interactive areas haven't been specifically hand-crafted for the classification task. In future work, we will explore how to train models that label patches spanning multiple heterogenous colonies. Furthermore, we will explore how the axis of interaction shapes the distribution of morphologically relevant features across colonies.

8. REFERENCES

[1] N. S. Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992.

[2] Y. Bengio. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

[3] J. Bouvrie. Notes on convolutional neural networks. Technical report, 2006.

[4] D. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber. Deep neural networks segment neuronal membranes in electron microscopy images. In *Advances in neural information processing systems*, pages 2843–2851, 2012.

[5] S. Falkow, R. Isberg, and D. Portnoy. The interaction of bacteria with mammalian cells. *Annual review of cell biology*, 8(1):333–363, 1992.

[6] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[7] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-6(6):721–741, Nov 1984.

[8] S. Godbole and S. Sarawagi. Discriminative methods for multi-labeled classification. In *Advances in Knowledge Discovery and Data Mining*, pages 22–30. Springer, 2004.

[9] R. Goodacre, R. Burton, N. Kaderbhai, A. M. Woodward, D. B. Kell, P. J. Rooney, et al. Rapid identification of urinary tract infection bacteria using hyperspectral whole-organism fingerprinting and artificial neural networks. *Microbiology*, 144(5):1157–1170, 1998.

[10] M. Helmstaedter, K. L. Briggman, S. C. Turaga, V. Jain, H. S. Seung, and W. Denk. Connectomic reconstruction of the inner plexiform layer in the mouse retina. *Nature*, 500(7461):168–174, 2013.

[11] G. Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

[12] G. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[13] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[14] P. Hiremath and P. Bannigidad. Automated gram-staining characterization of digital bacterial cell images. In *Proc. IEEE Internationall. Conf. on Signal and Image Processing ICSIP*, pages 209–211, 2009.

[15] P. Hiremath and B. Parashuram. Automatic identification and classification of bacilli bacterial cell growth phases. *IJCA special issue on Recent Trends in Image Processing and Pattern Recognition, RTIPPR*, 2010.

[16] P. S. Hiremath and P. Bannigidad. Automatic classification of bacterial cells in digital microscopic images. In *Proc. SPIE*, volume 7546, pages 754613–754613–6, 2010.

[17] V. Jain, J. F. Murray, F. Roth, S. Turaga, V. Zhigulin, K. L. Briggman, M. N. Helmstaedter, W. Denk, and H. S. Seung. Supervised learning of image restoration with convolutional networks. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[18] V. Jain and S. Seung. Natural image denoising with convolutional networks. In *Advances in Neural Information Processing Systems*, pages 769–776, 2009.

[19] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[20] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Computer Science Department, University of Toronto, Tech. Rep*, 1(4):7, 2009.

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

[22] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

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

[24] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 609–616. ACM, 2009.

[25] J. Liu, F. B. Dazzo, O. Glagoleva, B. Yu, and A. K. Jain. Cmeias: a computer-aided system for the image analysis of bacterial morphotypes in microbial communities. *Microbial Ecology*, 41(3):173–194, 2001.

[26] D. G. Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

[27] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814, 2010.

[28] M. Norouzi, M. Ranjbar, and G. Mori. Stacks of convolutional restricted boltzmann machines for shift-invariant feature learning. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2735–2742. IEEE, 2009.

[29] C. E. Rasmussen. The infinite gaussian mixture model. In *NIPS*, volume 12, pages 554–560, 1999.

[30] A. Saxe, P. W. Koh, Z. Chen, M. Bhand, B. Suresh, and A. Y. Ng. On random weights and unsupervised feature learning. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1089–1096, 2011.

[31] P. Smolensky. Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. chapter Information Processing in Dynamical Systems: Foundations of Harmony Theory, pages 194–281. MIT Press, Cambridge, MA, USA, 1986.

[32] K. Sohn, D. Y. Jung, H. Lee, and A. O. Hero. Efficient learning of sparse, distributed, convolutional feature representations for object recognition. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2643–2650. IEEE, 2011.

[33] S. Trattner, H. Greenspan, G. Tepper, and S. Abboud. Automatic identification of bacterial types using statistical imaging methods. *Medical Imaging, IEEE Transactions on*, 23(7):807–820, 2004.

[34] S. C. Turaga, J. F. Murray, V. Jain, F. Roth, M. Helmstaedter, K. Briggman, W. Denk, and H. S. Seung. Convolutional networks can learn to generate affinity graphs for image segmentation. *Neural Computation*, 22(2):511–538, 2010.