Computational Biology

Project 1: Automated Particle Picking in Cryo-EM

An Ju **Yangkun Zhang** **Tianxiao Shen**

Institute for Interdisciplinary Information Sciences

April, 2016

1 Introduction

This project is to pick particles from micrograph in cryo-electron microscopy (cryo-EM) [3]. Given a micrograph, our system will predict the center coordinates of particles in it.

We address this problem in a two-step procedure: first, we scan the whole micrograph by a sliding window and judge whether it contains a particle or not [2]; and then, for all the windows containing a particle, we compute their center and merge the ones nearby to get the final results.

An Ju programmed the major part. Yangkun Zhang assisted in coding and did program optimization, including efficient GPU implementation. Tianxiao Shen was in charge of algorithm design, report and demo. We did experiments and analysis together.

2 Algorithm

2.1 Window Classification

To judge whether a window contains a particle or not is a binary classification problem. We use a convolutional neural network (CNN) to deal with this task, which is one of the most powerful machine learning algorithms for image classification [1] .

2.1.1 Preprocessing

We preprocess all the values to make them range between 0 and 1, by subtracting min and then dividing by $\max - \min$. We directly train our network on these values.

2.1.2 Network Architecture

Our network contains 5 learned layers—3 convolutional and 2 fully-connected. The convolutional layers have 48 kernels of size 11×11 with stride 4, 128 kernels of size 5×5 with

stride 1, and 197 kernels of size 3×3 with stride 1 respectively. Each of them is followed by a max-pooling layer. The first fully-connected layer has 512 neurons, and the second has 2 neurons, corresponding to the two classes.

We use Rectified Linear Units (ReLUs) nonlinearity, and the final layer is fed to a 2-way softmax to produce a Bernoulli distribution. We use the cross-entropy between the true and predicted distribution as our loss function. And we use Stochastic Gradient Descent (SGD) with Nesterov momentum to train our model [4]. Each micrograph is a batch.

2.1.3 Training Data

As the two classes are skewed—most windows do not contain a particle, we adopt a simple strategy to get balanced training data: all golden particles are used as positive cases (each particle determines a window by locating its center), and we randomly sample negative cases of the same amount.

2.2 Merge Neighboring Windows

A particle could be contained by multiple overlapping windows, and thus we need to merge them into a single one. For a window predicted to have a particle in it, we compute the coordinates of its center and find its nearest neighbor among all the particles picked so far. If the distance between them is less than a threshold d, we merge them in the center of mass way; otherwise we regard it as a new particle.

We calculate the confidence of a particle by summing up its components' confidence minus 0.5, since 0.5 is the bias. If this value is greater than a threshold C, we report it as a final predicted particle.

The overall procedure is described in Algorithm 1.

3 Experiments

There are 40 micrographs in training data, from which we use 32 to train our network, and 8 for validation to tune the hyper-parameters. The results are based on 20 blind testing micrographs.

3.1 Parameters Setting

The window size is 200×200, and the stride is 15. Distance threshold $d = 30$. Confidence threshold $C = 1.65$.

3.2 Results

For window classification, our model achieves 93.54% accuracy. Table 1 shows the performance on test data. The average precision, recall, and F-measure are 59.33%, 74.26%, 64.46% respectively.

Algorithm 1 Particle Picking

Input: a micrograph g

Output: a list l of particles

$l, \hat{l} \leftarrow \emptyset$
for all window w in g **do**
  $(p_0, p_1) \leftarrow \mathrm{CNN}(w)$
  if $p_1 > 0.5$ **then**
    $(x_w, y_w) \leftarrow \mathrm{center}(w)$
    $(x, y, m, c) \leftarrow \mathrm{findNearestNeighbor}((x_w, y_w), \hat{l})$
    if $\mathrm{dist}((x_w, y_w), (x, y)) < d$ **then**
      $(x, y) \leftarrow (\frac{mx+x_w}{m+1}, \frac{my+y_w}{m+1})$
      $(m, c) \leftarrow (m + 1, c + p_1 - 0.5)$
    else
      $\hat{l} \leftarrow \hat{l} \cup \{(x_w, y_w, 1, p_1 - 0.5)\}$
    end if
  end if
end for
for all $(x, y, m, c) \in \hat{l}$ **do**
  if $c > C$ **then**
    $l \leftarrow l \cup \{(x, y)\}$
  end if
end for

Index	1	2	3	4	5	6	7	8	9	10
Precision	64.96	57.19	67.77	53.33	49.47	70.00	72.00	66.94	60.53	23.25
Recall	83.33	96.62	86.42	71.15	86.96	56.44	64.29	42.94	85.21	70.89
F-measure	73.01	71.85	75.97	60.97	63.06	62.49	67.92	52.32	70.78	35.01
Index	11	12	13	14	15	16	17	18	19	20
Precision	58.30	69.48	58.42	57.84	59.80	74.74	71.74	23.59	58.21	69.09
Recall	87.65	87.76	84.66	73.76	85.55	69.89	54.34	38.60	84.38	74.47
F-measure	70.03	77.56	69.13	64.84	70.39	72.24	61.84	29.28	68.89	71.68

Table 1: Precision, recall, and F-measure on test data.

4 Discussion

4.1 Contaminant

Our model performs poorly on micrograph 10 and 18. We find that they have a large area of contaminants, which lead to numerous false positives (see figure 1).

Follow [2], we tried to do Principal Component Analysis (PCA) over the power spectra of extracted windows, and eliminate outliers according to their distribution. But it did not work :(. We think a procedure of edge detection, connected components labeling and convex hull computation [5] might be a good solution to detect contaminants.

4.2 Boundary

When we tested on validation data, we realized that we did not consider windows lying out. Therefore, we failed to pick out particles on the boundary (see figure 2).

We tried various padding methods to take these incomplete windows into consideration, including padding with mean, Gaussian noise, and folding. But none of them worked :(. Nevertheless, it seems that these particles do not affect evaluation much.

Acknowledgments

We thank Zhipeng Jia for the helpful discussion with him.

References

[1] 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.

[2] R. Langlois, J. Pallesen, J. T. Ash, D. N. Ho, J. L. Rubinstein, and J. Frank. Automated particle picking for low-contrast macromolecules in cryo-electron microscopy. *Journal of structural biology*, 186(1):1–7, 2014.

[3] M. Liao, E. Cao, D. Julius, and Y. Cheng. Structure of the trpv1 ion channel determined by electron cryo-microscopy. *Nature*, 504(7478):107–112, 2013.

[4] I. Sutskever, J. Martens, G. Dahl, and G. Hinton. On the importance of initialization and momentum in deep learning. In *Proceedings of the 30th international conference on machine learning (ICML-13)*, pages 1139–1147, 2013.

[5] Y. Zhu, B. Carragher, and C. S. Potter. Contaminant detection: improving template matching based particle selection for cryoelectron microscopy. In *Biomedical Imaging: Nano to Macro, 2004. IEEE International Symposium on*, pages 1071–1074. IEEE, 2004.

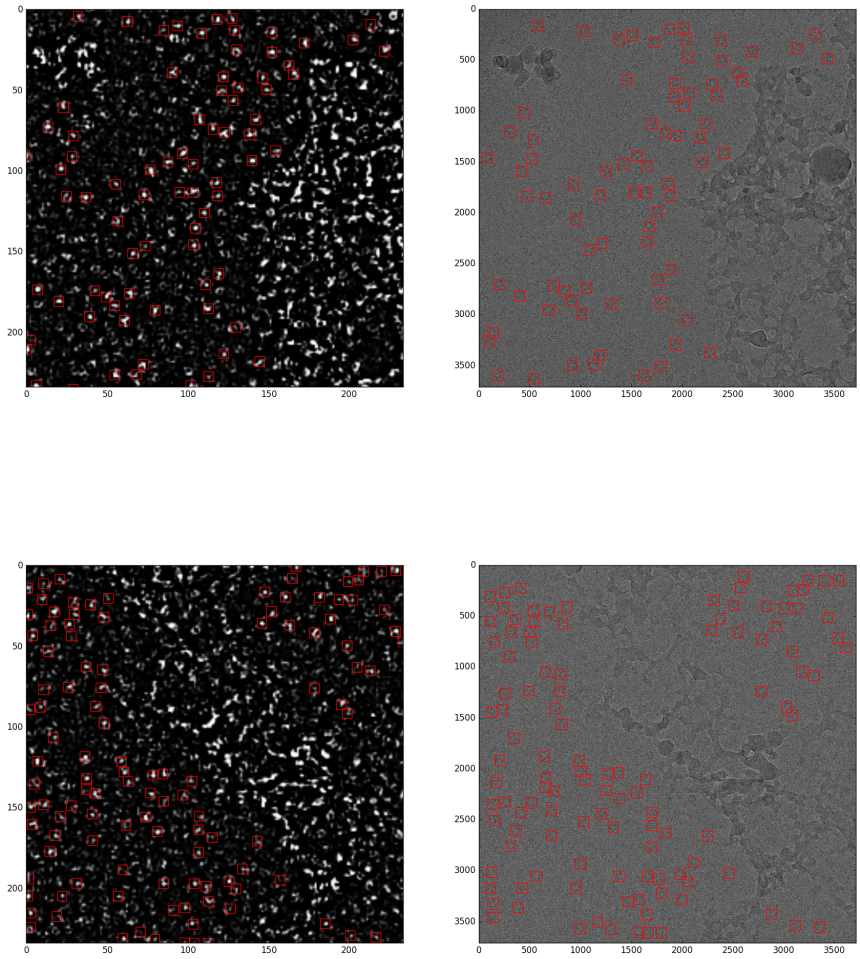

Figure 1: Micrograph 10 and 18 in test data. Images on the left show our network's output confidence (probability of containing a particle). Images on the right are original micrographs. Red windows contain golden particles as their center.

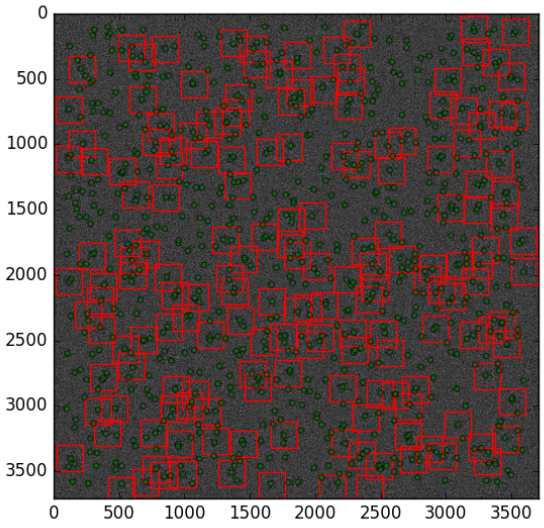

Figure 2: Micrograph 8 in validation data (i.e. micrograph 40 in all training data). Green points are predicted particles. Red windows contain golden particles as their center. On the bottom there are several windows lying out, which we missed.