Automated Detection and Sorting of Microencapsulation via Machine Learning

Albert Chu, Du Nguyen, Sachin S. Talathi, Aaron C. Wilson, Congwang Ye, William L. Smith, Alan D. Kaplan, Eric B. Duoss, Joshua K. Stolaroff, and Brian Giera

Supplementary Materials

Machine learning algorithm architecture

Our Convolutional Neural Network (CNN) [1] configuration follows a minimalist design in comparison to other models we explored, as shown in **Figure SM1**. The model contains a total of 6 weight layers (4 convolution and 2 fully-connected layers). Convolutional kernels are of size 3×3 with 16 output filters in the first two convolution blocks and 32 output filters in the third and fourth convolution blocks. Layer inputs are normalized via batch normalization[2] with a batch size of 64. All layers with the exception of the output layer uses ReLU nonlinear activation[1,3], whereas the output layer uses softmax activation[4] over four categories. Merge pooling layers follow each convolution layer, which consist of adding the average-pooling and max-pooling layers; pooling kernel sizes are 2×2 with stride of (2, 2). The fully-connected layers are flattened with the penultimate layer implementing dropout[5] with rate of 0.5 during training. Since our input data layer receives only one channel images of size 32×32, we use the open-source python package OpenCV[6] to convert all camera images to grayscale and resizing them to the proper dimensions.

We also explored VGG[7] and Inception[2], which are deeper layered networks than what we describe above. On those networks, pre-training on ImageNet[8] provided no advantages due to domain mismatch[9], since the source domain, i.e. ImageNet, is incongruent with our target domain of microencapsulation images. In terms of performance, the complex models with larger input sizes were outperformed by the simple model (139×139 vs. 32×32). Admittedly, this comparison is based on a modest training set of 74,000 images and larger networks will tend to overfit on smaller training sets. Nonetheless, the biggest factor in choosing our simple model is the training time due to the difference in number of parameters and input size. Our simple network yields 80k parameters, and 1.25M multiply-accumulate operations (MACs), compared with 138M parameters and 15.5B MACs for VGG and 11M parameters and 1.5M MACs for Inception.

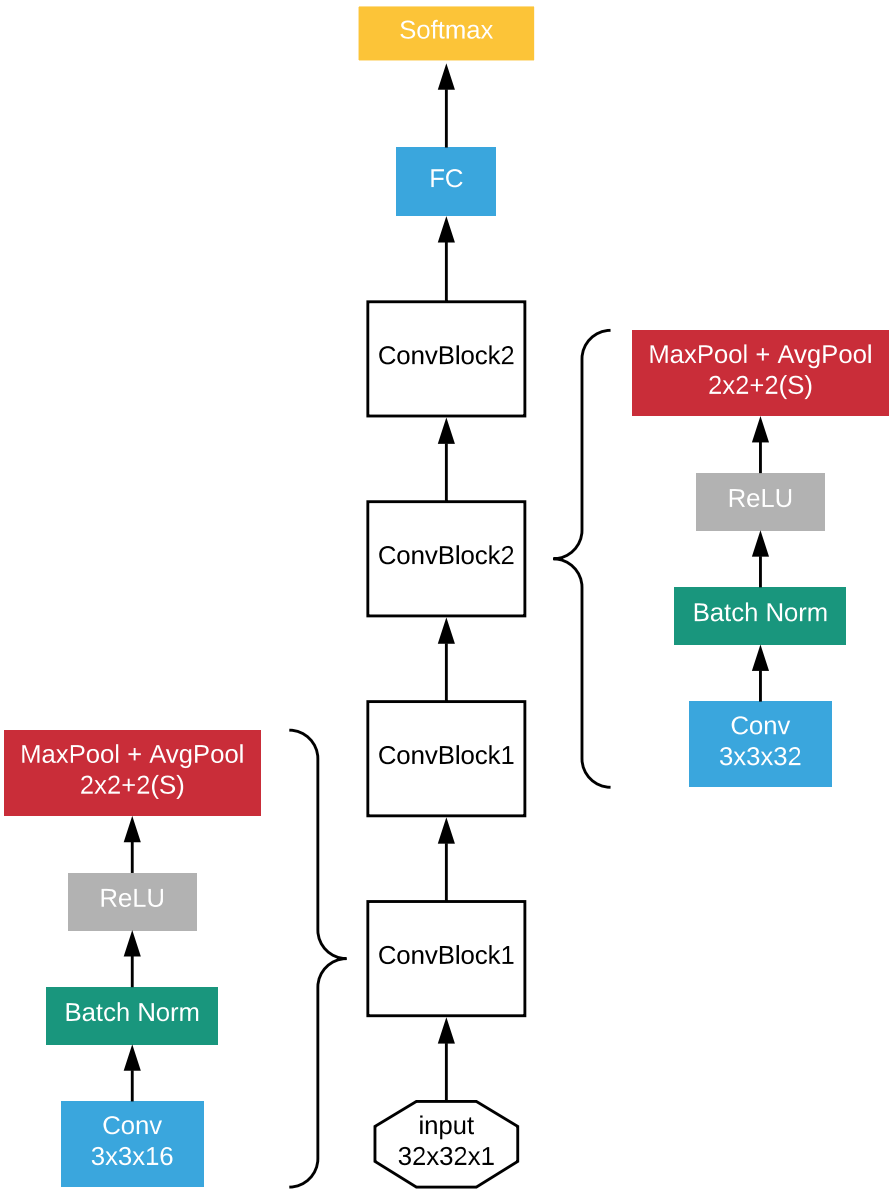

Figure SM1. Machine learning model architecture schematic.

Sorting simulation sequences

The displays the sequence of events that contains uninterrupted non-dripping (0-10 and ~44-94 seconds) events, periods of moderate (~10-25 seconds) and high (~25-30 seconds) frequency fluctuations, and uninterrupted dripping events elsewhere. **Figure SM2** shows this sequence for three example models with different F1 scores and overlays the false collection, false rejection, and operator notification events. We identify with markers the algorithm-predicted class during false collection of non-dripping events or false rejection of dripping events. Misclassifications are evident in cases where markers do not coincide with the ground truth. We also highlight the time(s) where the valve controller sends a text message to the operator, which occurs when the image detections classifies 10 seconds of uninterrupted non-dripping events.

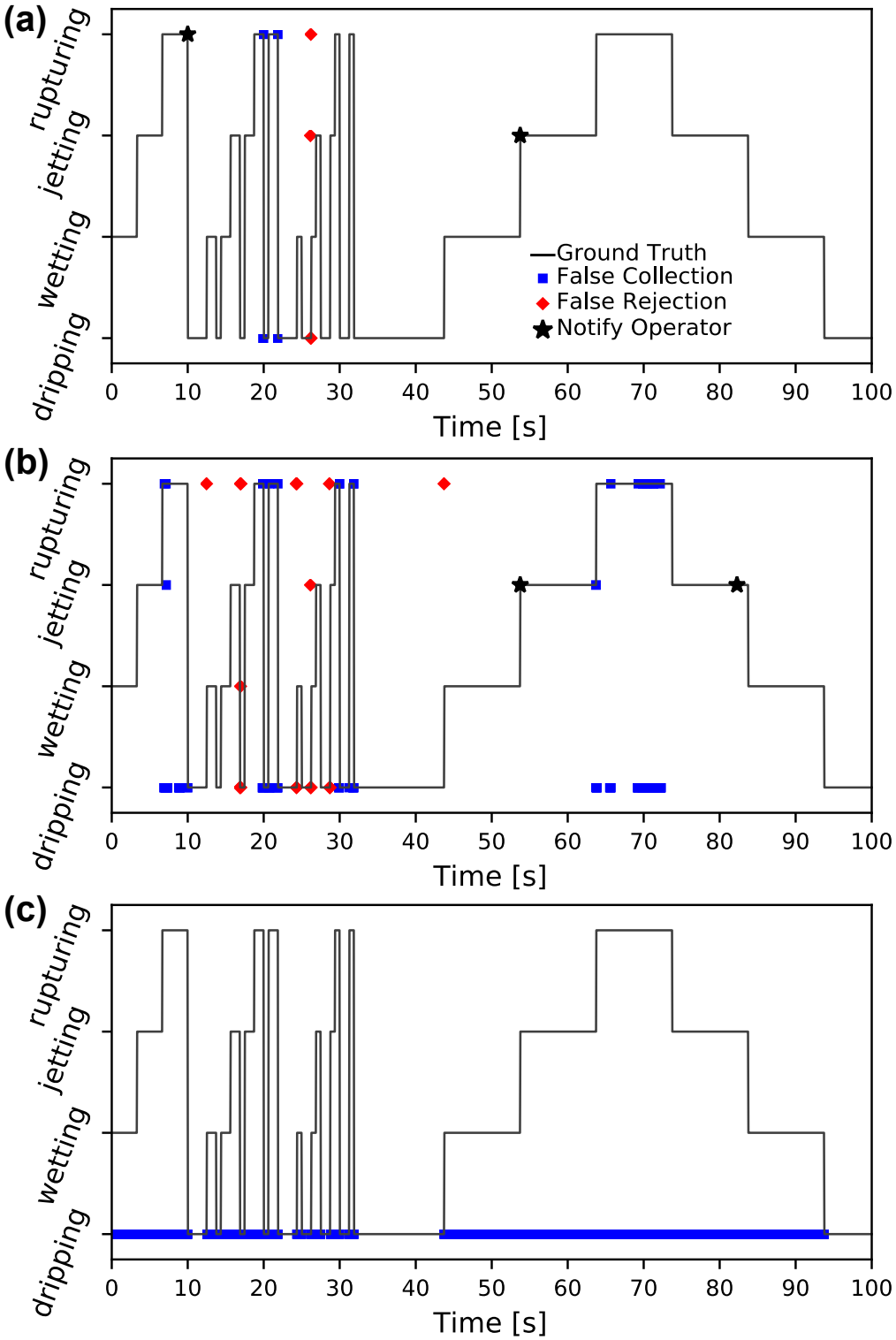

Figure SM2. Sorting simulations show the valving system's effectiveness for several image detection algorithms with varying prediction accuracy. Markers for false collection, false rejection, and operator notification events (legend) appear at the CNN-predicted class for models with F1-scores of 0.946 (a), 0.852 (b), and 0.220 (c). Per **Figure 4** in main text, the fraction of erroneous sorting events generally decreases with increasing F1-score and training set size.

References

1 A. Krizhevsky, I. Sutskever and G. E. Hinton, in *Advances in Neural Information Processing Systems 25*, eds. F. Pereira, C. J. C. Burges, L. Bottou and K. Q. Weinberger, Curran Associates, Inc., 2012, pp. 1097–1105.

2 S. Ioffe and C. Szegedy, *ArXiv150203167 Cs*.

3 V. Nair and G. E. Hinton, 8.

4 C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*, Springer-Verlag New York, Inc., 2006.

5 N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, 30.

6 G. Bradski and A. Kaehler, *Learning OpenCV: Computer Vision with the OpenCV Library*, O'Reilly Media, Inc., 2008.

7 K. Simonyan and A. Zisserman, *ArXiv14091556 Cs*.

8 J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li and Li Fei-Fei, IEEE, 2009, pp. 248–255.

9 Z. Shen, Z. Liu, J. Li, Y.-G. Jiang, Y. Chen and X. Xue, IEEE, 2017, pp. 1937–1945.