Artificial Neural Networks for Sample Recognition: Datasets, Architectures, Performance

Martin Savko
savko@synchrotron-soleil.fr

Joint ISPyB and MXCuBE developers meeting, ESRF, Grenoble 17th May 2022

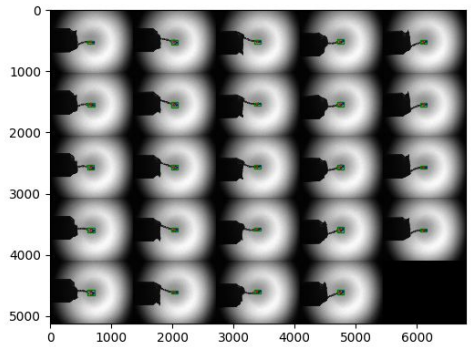
Outline

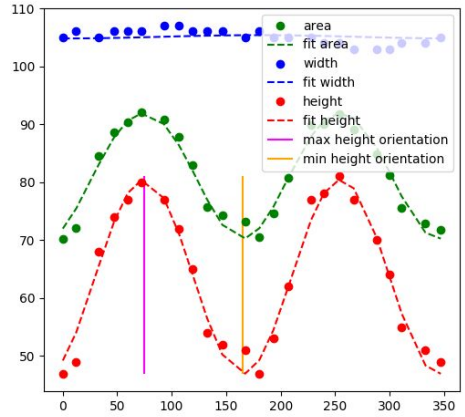
- Problem: why are we interested?
- Datasets: how do we measure?
- Architectures: what is our hypothesis space?
- Performance: how do we measure it? How robust is it?
- Outlook

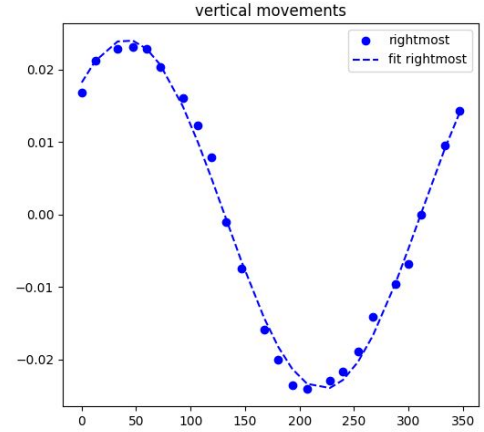
Automated sample alignment -- current method

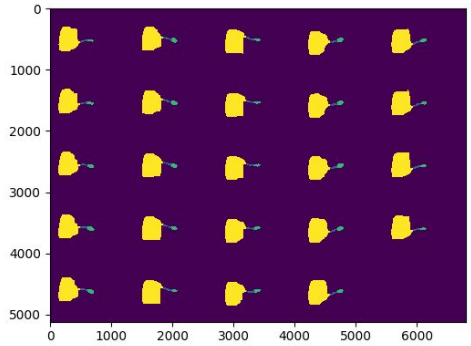
- detect loop, pin and stem by analysis of variance and sum curves
- code parallelized ~ run time about 8 seconds on multicore machine
- outputs segmented images
- models bounding box projection, centroid and rightmost point
- E.g. we know at what angle there is projected area minimal and maximal -- useful for efficient raster scanning.
- Works most of the time but sometimes fails miserably
    - success crucially dependent on good background model

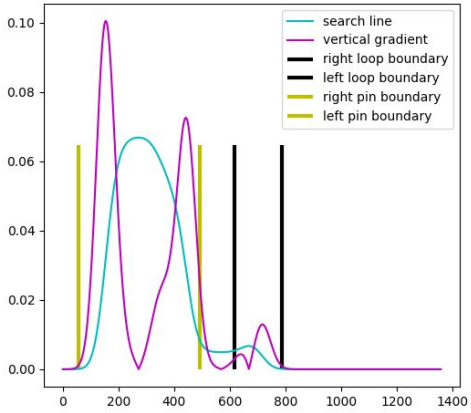
/nfs/data3/2020_Run3/com-proxima2a/Commissioning/2020-07-27/ICE/CPS-2597/2/optical/2_1

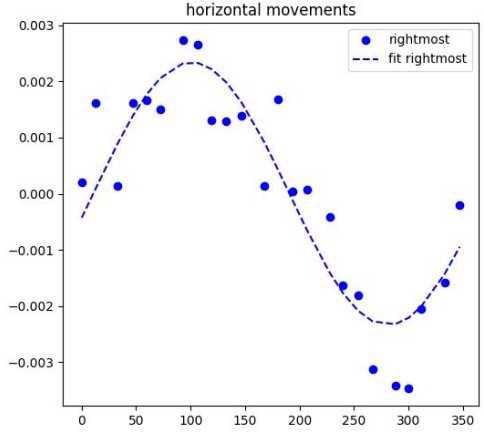




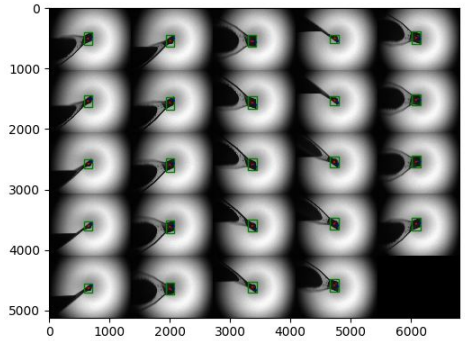


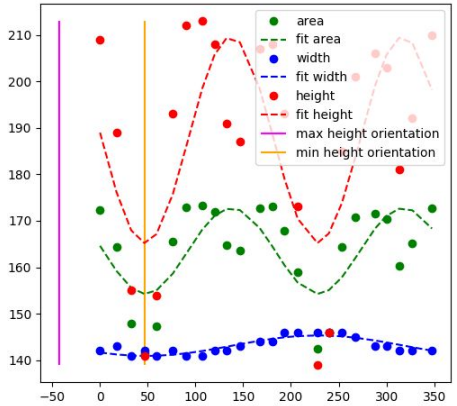


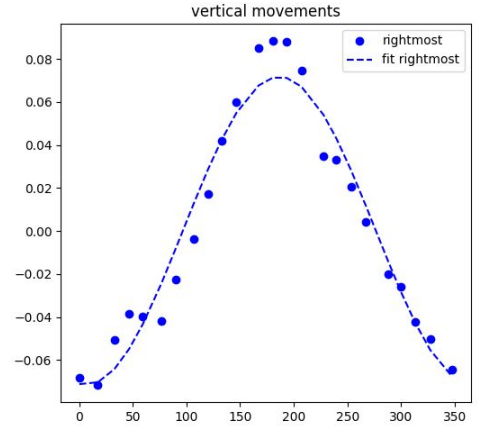


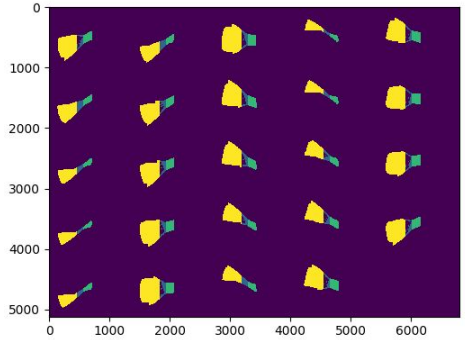


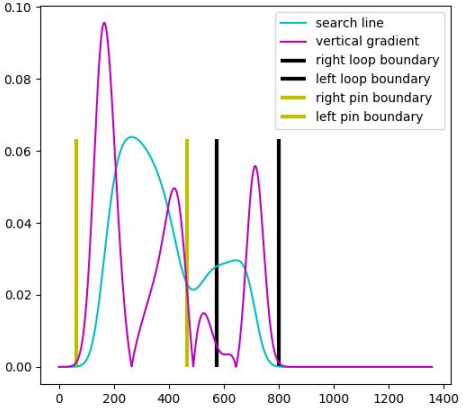
/nfs/data3/2020_Run3/com-proxima2a/Commissioning/2020-07-27/ICE/CPS-2389/17/optical/17_1

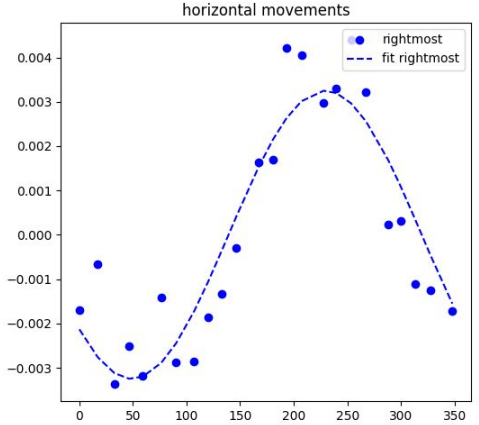




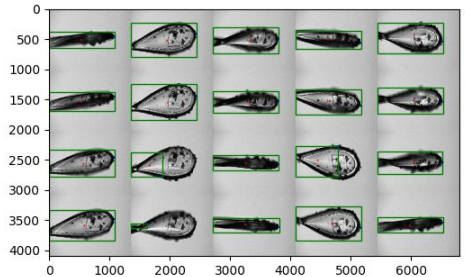


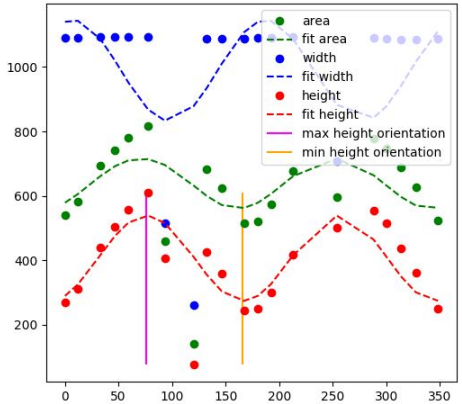


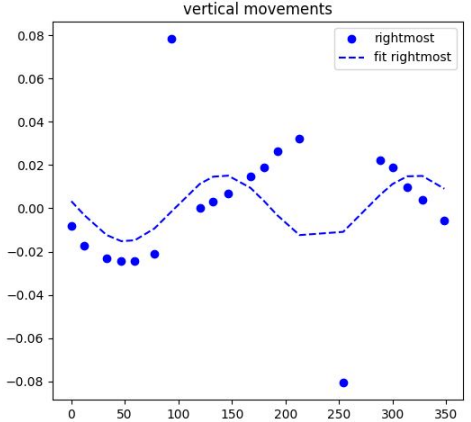


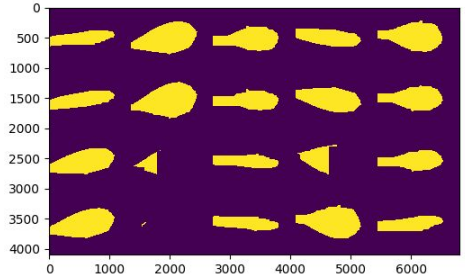


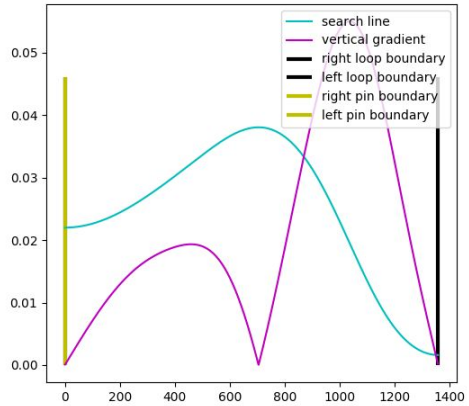
/nfs/data3/2020_Run3/com-proxima2a/Commissioning/2020-07-27/ICE2/CPS-3003/3/optical/3_2

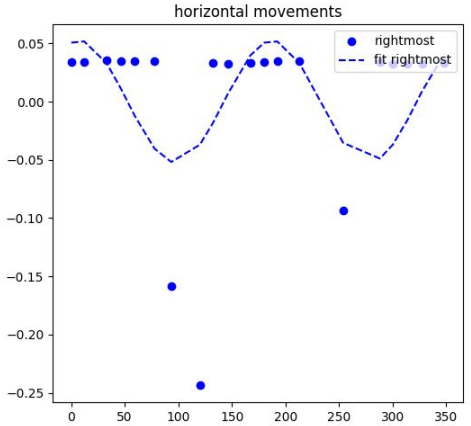








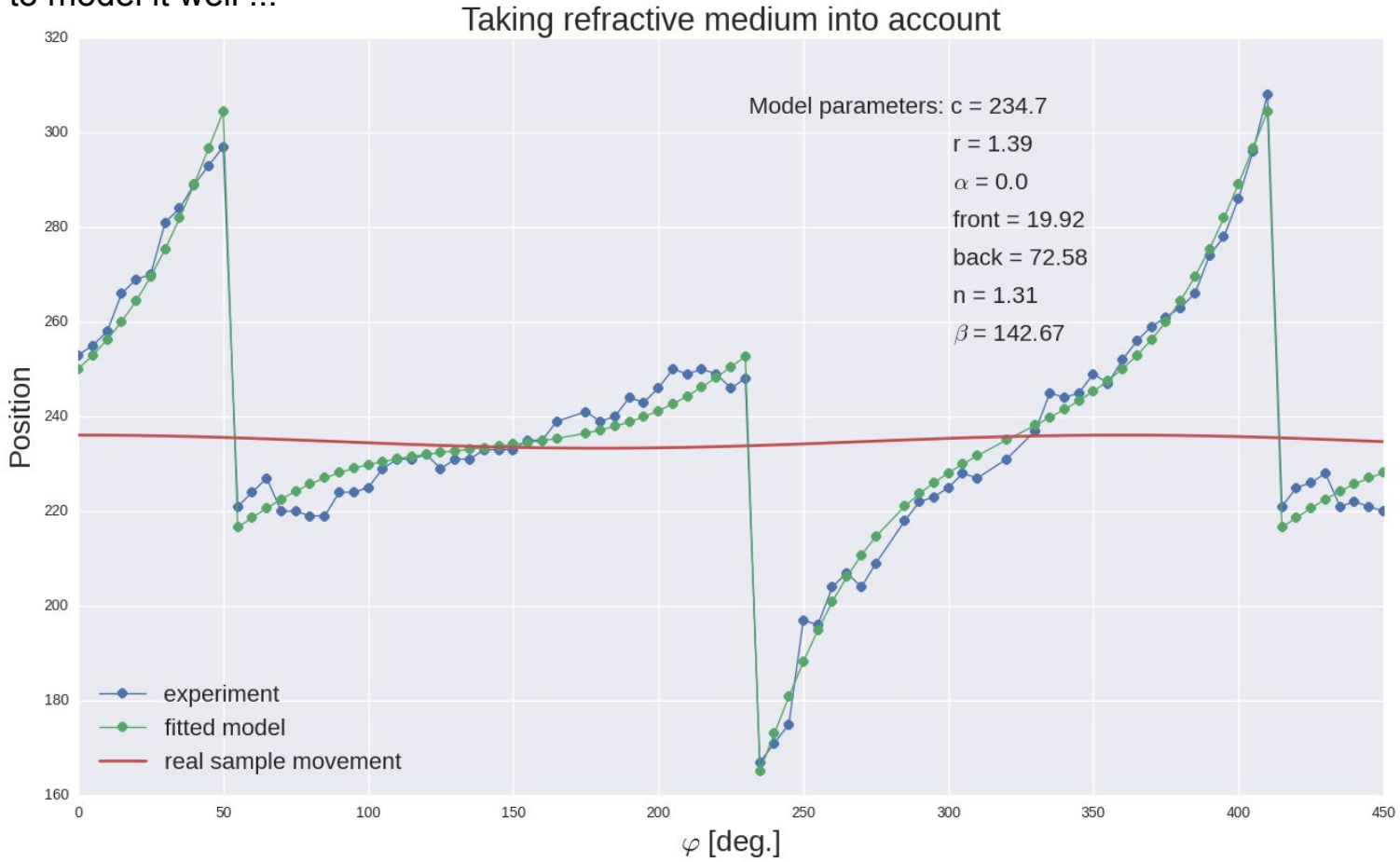




We sometimes need more accurate models of sample image movement …

- although sample moves on a circle as omega axis changes, its image almost never does (only if there is just material of refractive index of 1 around it). It follows much stranger law. Law nonetheless -- one just needs more parameters to model it …

This is example of a sample aligned almost perfectly. It's image is moving across many microns … it is important to model it well ...



Slab model

$$i = arcsin(sin(\frac{\varphi}{n})) - \beta$$

$$y_{corr} = y - \frac{front * sin(\varphi - i)}{cos(i)}$$

$$y_{corr} = y - \frac{back * sin(-\varphi - i)}{cos(i)}$$

What we want is system that:

- recognizes different sample components
  - crystal, loop, stem, pin, ice, dust, mother liquor
  - pixelwise
- works at arbitrary scale
- is fast

But to begin with, we need a benchmark!

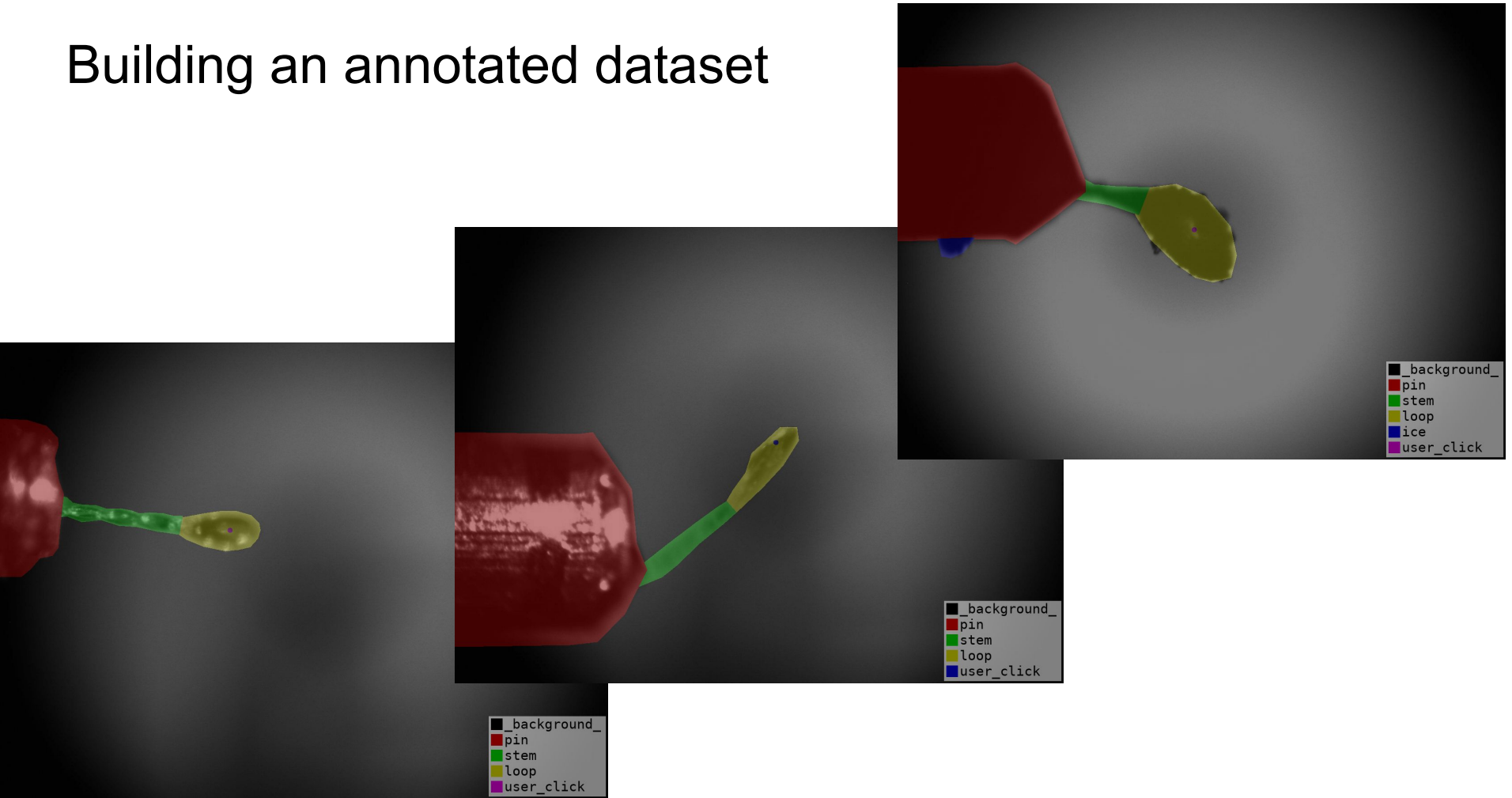
Datasets

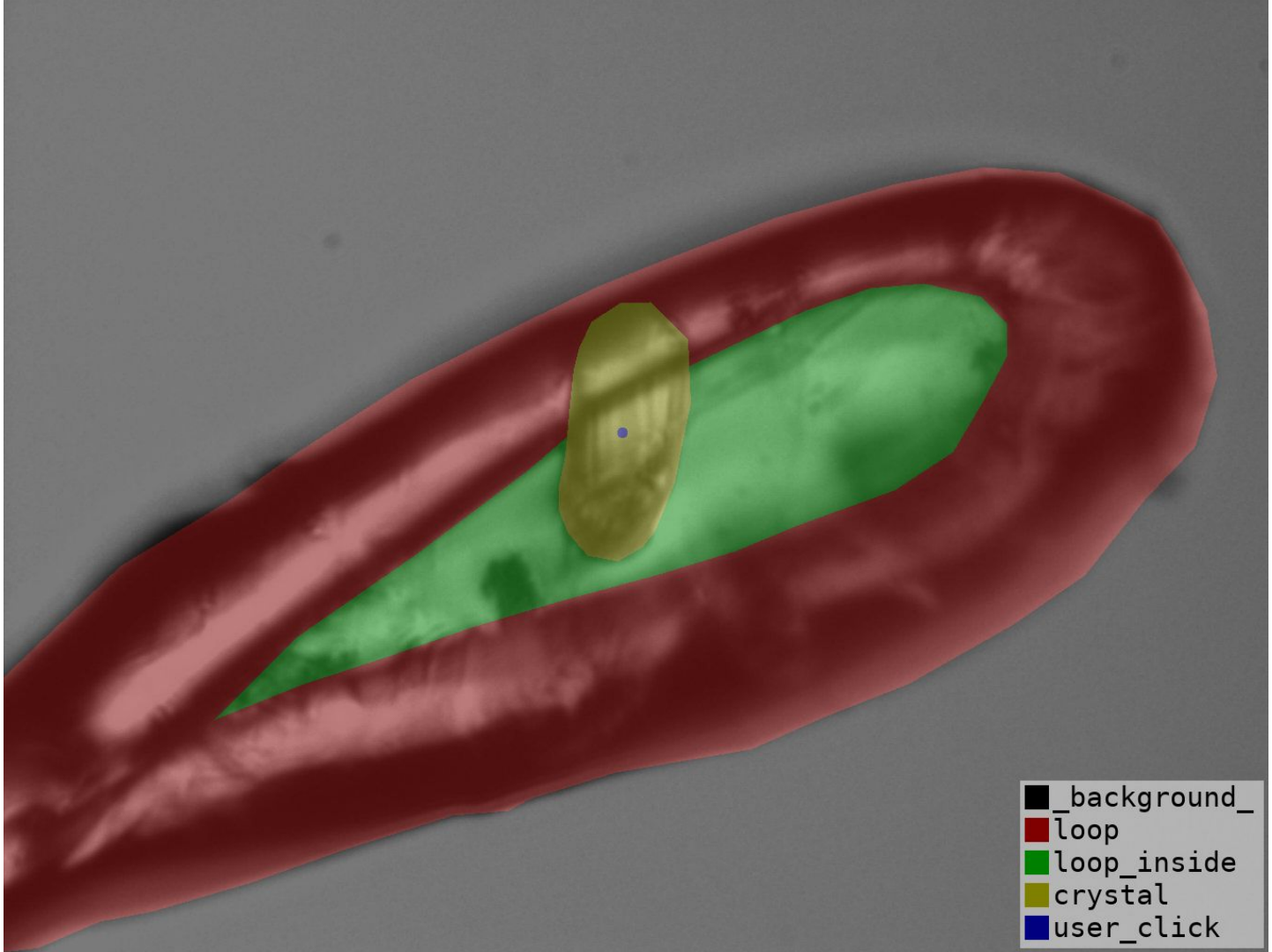
- Video streams recorded during sample alignment
  - 73301 alignments, ~65 images in each
- Click dataset
  - 525312 images with associated clicks
  - either clicks during alignments or sample exploration i.e. double clicks
- Semantically segmented dataset
  - Pixelwise annotated dataset
  - 1248 images in the database

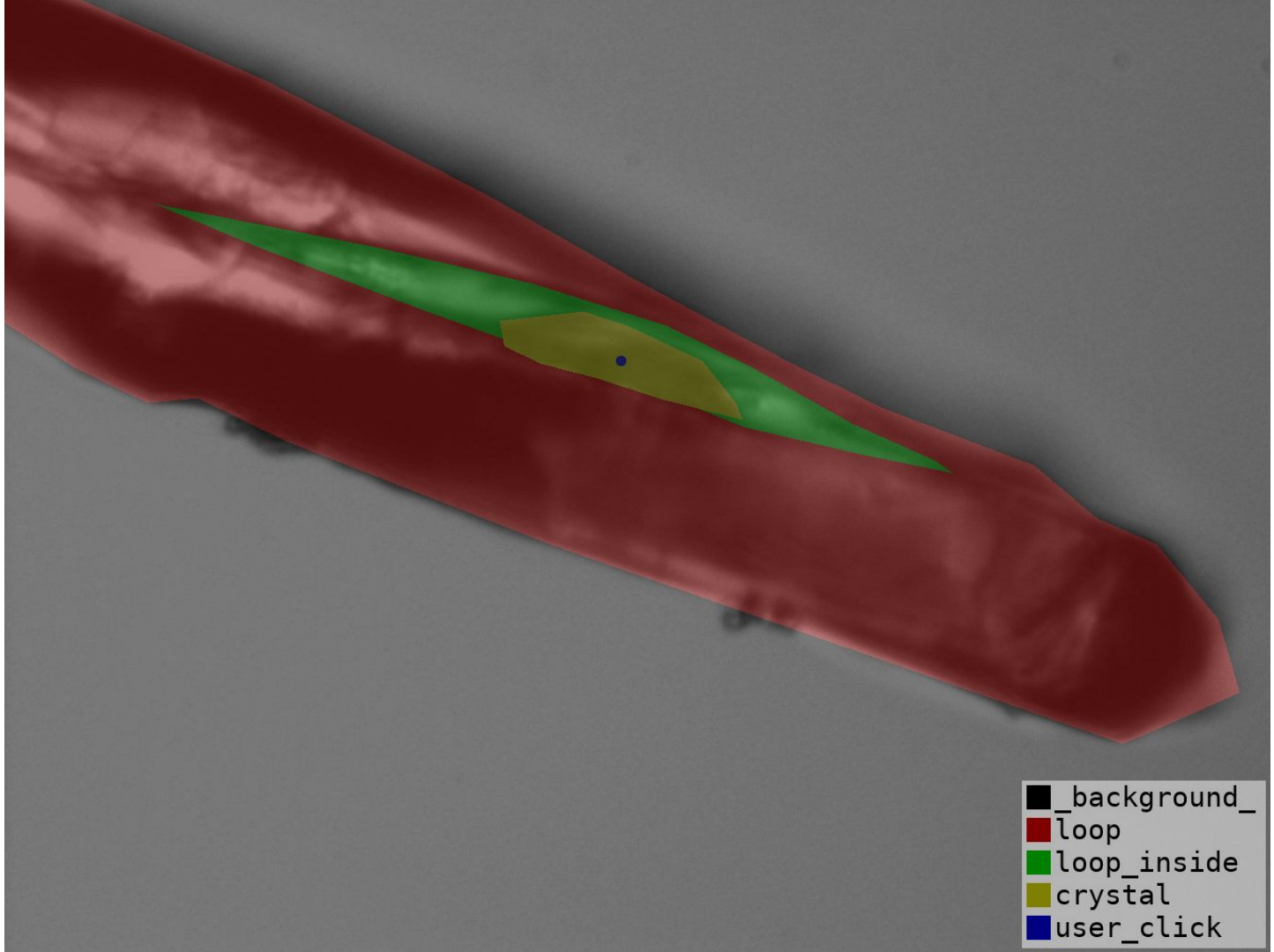
Segmentation Dataset

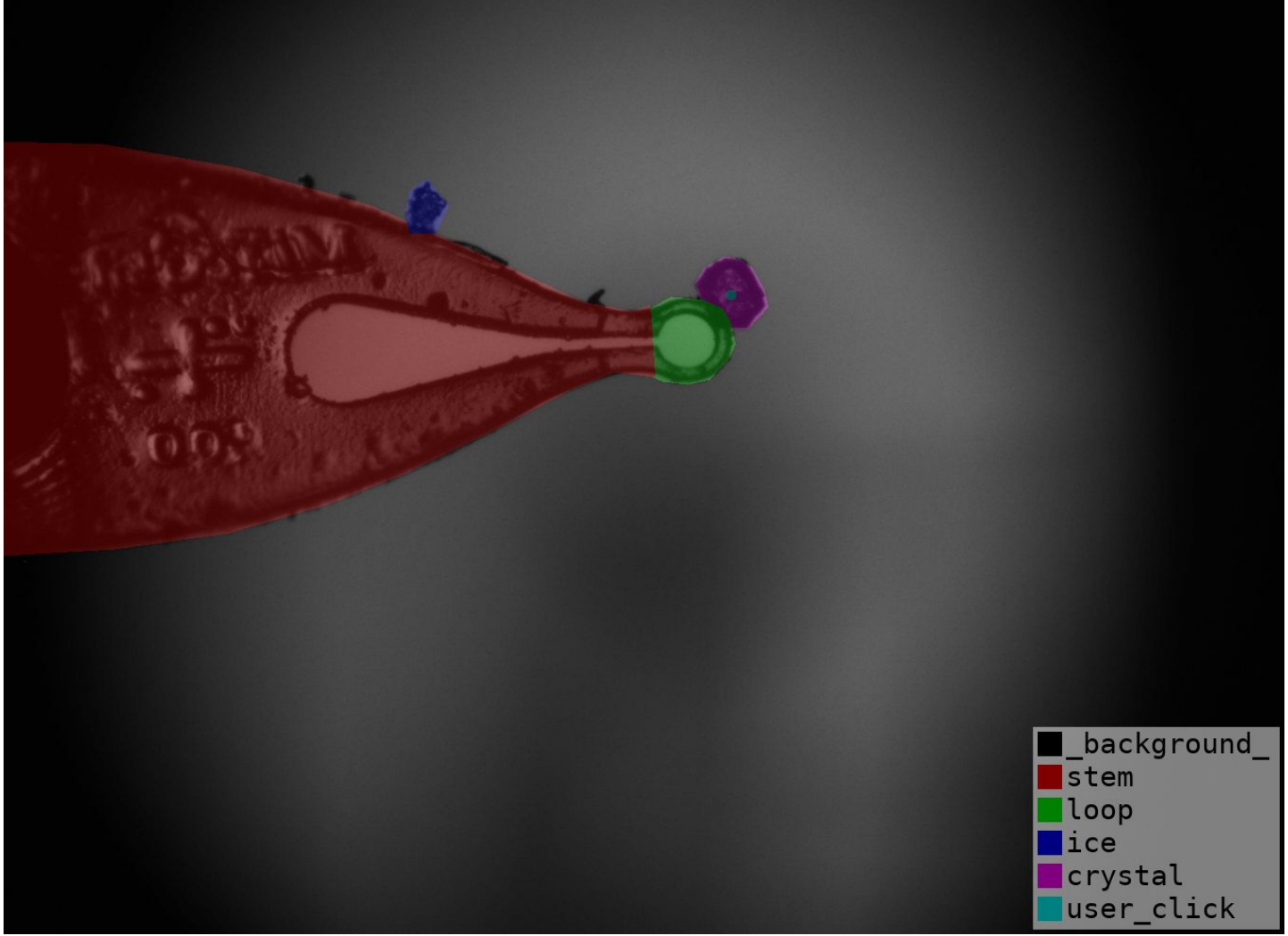
- pixelwise annotation is costly
  - only 1248 images
  - notions: ['crystal', 'loop_inside', 'loop', 'stem', 'pin', 'capillary', 'ice', 'foreground', 'click']
- can we use cheaper kind of annotation?
  - user clicks: ~500K images in the database
  - video stream: ~2M images in the database
  - images acquired during auto centring using prior, shallow, system (~30K images)

Building an annotated dataset



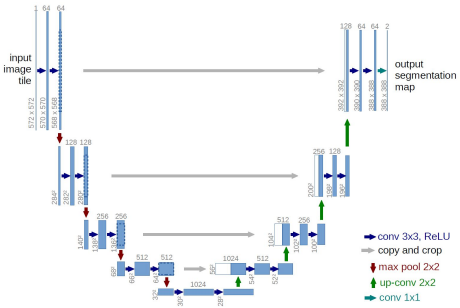

background
loop
loop_inside
crystal
user_click


background
loop
loop_inside
crystal
user_click


background
stem
loop
ice
crystal
user_click

Network Architecture

- U-net based
  - arXiv:1505.04597
  - arXiv:1611.09326
  - arXiv:1610.02357



U-Net: Convolutional Networks for Biomedical Image Segmentation

Olaf Ronneberger, Philipp Fischer, and Thomas Brox

Computer Science Department and BIOSS Centre for Biological Signalling Studies, University of Freiburg, Germany
ronneber@informatik.uni-freiburg.de,
WWW home page: http://lmb.informatik.uni-freiburg.de/

Abstract. There is large consent that successful training of deep networks requires many thousand annotated training samples. In this paper, we present a network and training strategy that relies on the strong use of data augmentation to use the available annotated samples more efficiently. The architecture consists of a contracting path to capture context and a symmetric expanding path that enables precise localization. We show that such a network can be trained end-to-end from very few images and outperforms the prior best method (a sliding-window network) on the ISBI challenge for segmentation of neu-es in electron microscopic stacks. Using the same net-on transmitted light microscopy images (phase contrast won the ISBI cell tracking challenge 2015 in these cate-rge margin. Moreover, the network is fast. Segmentation mage takes less than a second on a recent GPU. The full n (based on Caffe) and the trained networks are available .informatik.uni-freiburg.de/people/ronneber/u-net.

Xception: Deep Learning with Depthwise Separable Convolutions

François Chollet
Google, Inc.
fchollet@google.com

Abstract

We present an interpretation of Inception modules in convolutional neural networks as being an intermediate step in-between regular convolution and the depthwise separable convolution operation (a depthwise convolution followed by a pointwise convolution). In this light, a depthwise separable convolution can be understood as an Inception module with a maximally large number of towers. This observation leads us to propose a novel deep convolutional neural network architecture inspired by Inception, where Inception modules have been replaced with depthwise separable convolutions. We show that this architecture, dubbed Xception, slightly outperforms Inception V3 on the ImageNet dataset (which Inception V3 was designed for), and significantly outperforms Inception V3 on a larger image classification dataset comprising 350 million images and 17,000 classes. Since the Xception architecture has the same number of parameters as Inception V3, the performance gains are not due to increased capacity but rather to a more efficient use of model parameters.

as GoogLeNet (Inception V1), later refined as Inception V2 [7], Inception V3 [21], and most recently Inception-ResNet [19]. Inception itself was inspired by the earlier Network-In-Network architecture [11]. Since its first introduction, Inception has been one of the best performing family of models on the ImageNet dataset [14], as well as internal datasets in use at Google, in particular JFT [5].

The fundamental building block of Inception-style models is the Inception module, of which several different versions exist. In figure 1 we show the canonical form of an Inception module, as found in the Inception V3 architecture. An Inception model can be understood as a stack of such modules. This is a departure from earlier VGG-style networks which were stacks of simple convolution layers.

While Inception modules are conceptually similar to convolutions (they are convolutional feature extractors), they empirically appear to be capable of learning richer representations with less p
how do they differ fro
strategies come after I

1.1. The Inception

The One Hundred Layers Tiramisu: Fully Convolutional DenseNets for Semantic Segmentation

Simon Jégou[1] Michal Drozdzal[2,3] David Vazquez[1,4] Adriana Romero[1] Yoshua Bengio[1]
[1]Montreal Institute for Learning Algorithms [2]École Polytechnique de Montréal
[3]Imagia Inc., Montréal, [4]Computer Vision Center, Barcelona
simon.jegou@gmail.com, michal@imagia.com, dvazquez@cvc.uab.es,
adriana.romero.soriano@umontreal.ca, yoshua.umontreal@gmail.com

Abstract

State-of-the-art approaches for semantic image segmentation are built on Convolutional Neural Networks (CNNs). The typical segmentation architecture is composed of (a) a downsampling path responsible for extracting coarse semantic features, followed by (b) an upsampling path trained to recover the input image resolution at the output of the model and, optionally, (c) a post-processing module (e.g. Conditional Random Fields) to refine the model predictions.

Recently, a new CNN architecture, Densely Connected Convolutional Networks (DenseNets), has shown excellent results on image classification tasks. The idea of DenseNets is based on the observation that if each layer is directly connected to every other layer in a feed-forward fashion then the network will be more accurate and easier to train.

In this paper, we extend DenseNets to deal with the problem of semantic segmentation. We achieve state-of-the-art results on urban scene benchmark datasets such as CamVid and Gatech, without any further post-processing module nor pretraining. Moreover, due to smart construction of the model, our approach has much less parameters than currently published best entries for these datasets. Code to reproduce the experiments is publicly available here : https://github.com/SimJeg/FC-DenseNet

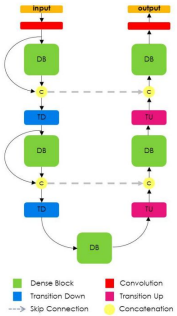


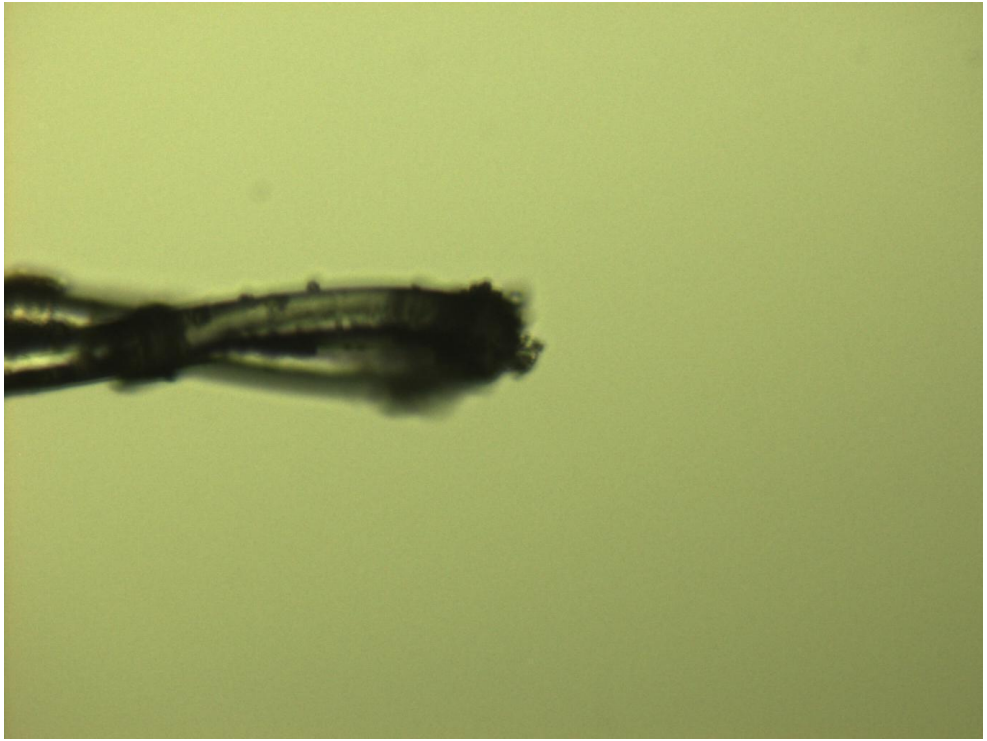
Figure 1. Diagram of our architecture for semantic segmentation. Our architecture is built from dense blocks. The diagram is composed of a downsampling path with 2 Transitions Down (TD) and an upsampling path with 2 Transitions Up (TU). A circle repre-

```
Total params: 3,494,008
Trainable params: 3,390,936
Non-trainable params: 103,072
```

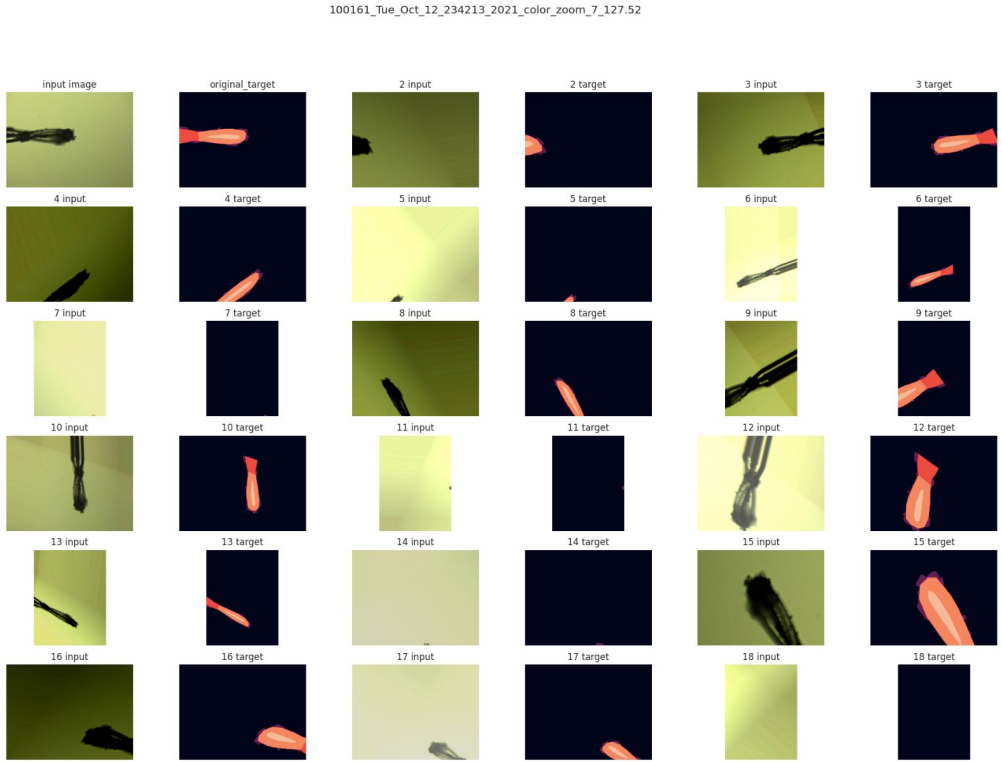
Training

- TensorFlow used through Keras
- Optimization algorithm is RMSProp, with initial learning rate of 0.001
- Training for 70 epochs. Each epoch exposes ~10000 sample images
- Dynamic learning rate decay, Reducing learning rate by factor of 2 after 3 epoch without drop in validation loss
- Regularization Dropout of 0.2 after every layer, weight decay of 1e-4
- Heavy data augmentation
  - random flips and transpositions
  - random affine transforms (shift, shear, zoom, rotation)
  - random channel swaps and brightness modification
  - random background swaps (PX2A backgrounds at different illumination conditions, default PX1 beckgrounds)
- 2 x NVIDIA RTX 6000 (24 GB RAM)
- Mixed precision used during training for 3x speed up
  - 800 ms per batch
  - ~8 hours for model to converge

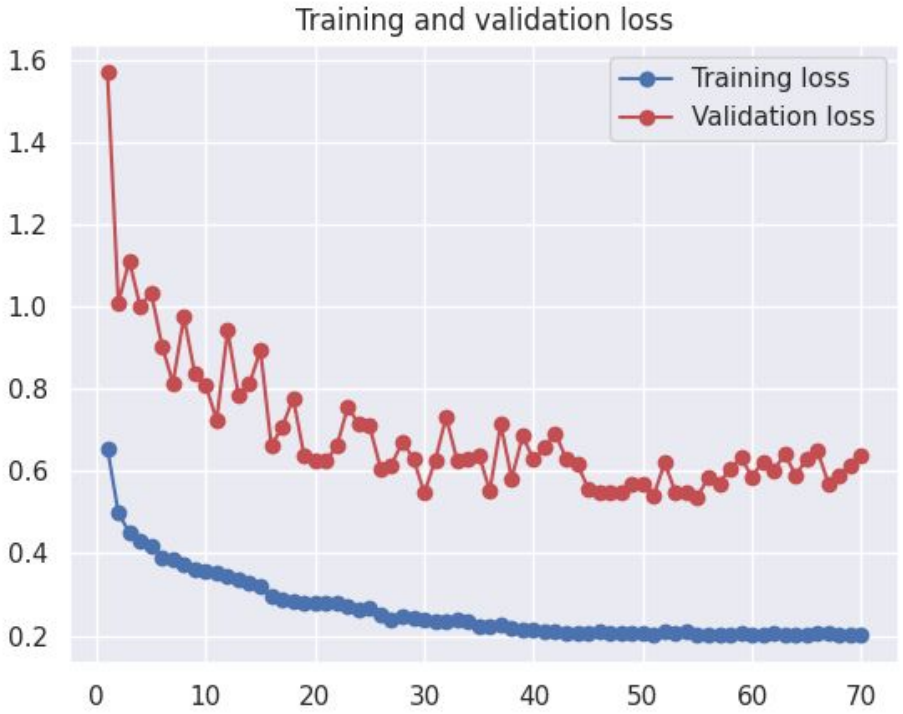
Data augmentation



Data augmentation

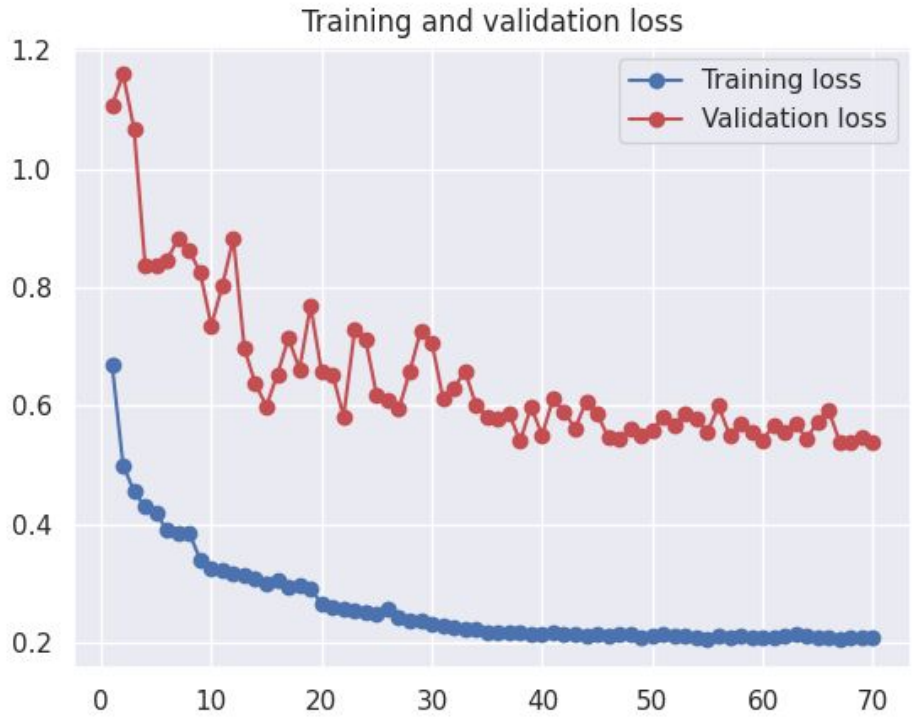


Training -- learning curves



using all images

Training (do we actually need more images?)



using half of images

Loss function and Metrics

- Focal loss (arXiv:1708.02002)
- Binary Intersection over Union as metrics

➢ True positive overlap (IoU) requirement (from 0.5 to 0.95)

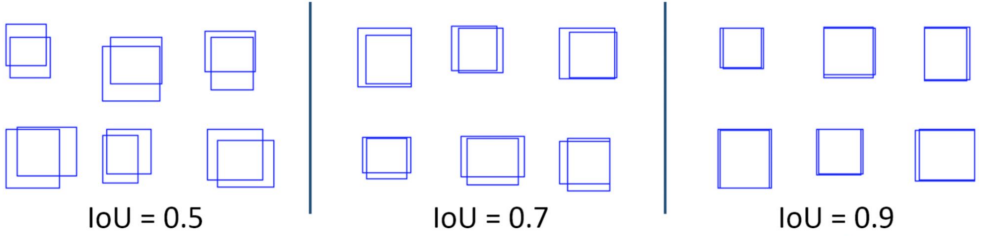


Figure credits: Dollár and Zitnick

Focal Loss for Dense Object Detection

Tsung-Yi Lin Priya Goyal Ross Girshick Kaiming He Piotr Dollár
Facebook AI Research (FAIR)

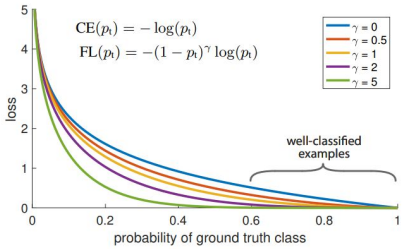


Figure 1. We propose a novel loss we term the *Focal Loss* that adds a factor $(1 - p_t)^{\gamma}$ to the standard cross entropy criterion. Setting $\gamma > 0$ reduces the relative loss for well-classified examples ($p_t > .5$), putting more focus on hard, misclassified examples. As our experiments will demonstrate, the proposed focal loss enables training highly accurate dense object detectors in the presence of vast numbers of easy background examples.

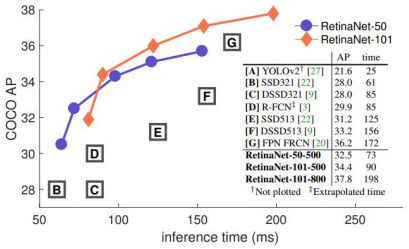


Figure 2. Speed (ms) versus accuracy (AP) on COCO `test-dev`. Enabled by the focal loss, our simple one-stage *RetinaNet* detector outperforms all previous one-stage and two-stage detectors, including the best reported Faster R-CNN [28] system from [20]. We show variants of RetinaNet with ResNet-50-FPN (blue circles) and ResNet-101-FPN (orange diamonds) at five scales (400-800 pixels). Ignoring the low-accuracy regime (AP<25), RetinaNet forms an upper envelope of all current detectors, and an improved variant (not shown) achieves 40.8 AP. Details are given in §5.

Abstract

The highest accuracy object detectors to date are based on a two-stage approach popularized by R-CNN, where a classifier is applied to a sparse *set of candidate object locations. In contrast, one-stage detectors that are applied over a regular,* dense *sampling of possible object locations have the potential to be faster and simpler, but have trailed the accuracy of two-stage detectors thus far. In this paper, we investigate why this is the case. We discover that the extreme foreground-background class imbalance encountered during training of dense detectors is the central cause. We propose to address this class imbalance by reshaping the standard cross entropy loss such that it down-weights the loss assigned to well-classified examples. Our novel* Focal Loss *focuses training on a sparse set of hard examples and prevents the vast number of easy negatives from overwhelming the detector during training. To evaluate the effectiveness of our loss, we design and train a simple dense detector we call RetinaNet. Our results show that when trained with the focal loss, RetinaNet is able to match the speed of previous one-stage detectors while surpassing the accuracy of all existing state-of-the-art two-stage detectors. Code is at:* `https://github.com/facebookresearch/Detectron`.

1. Introduction

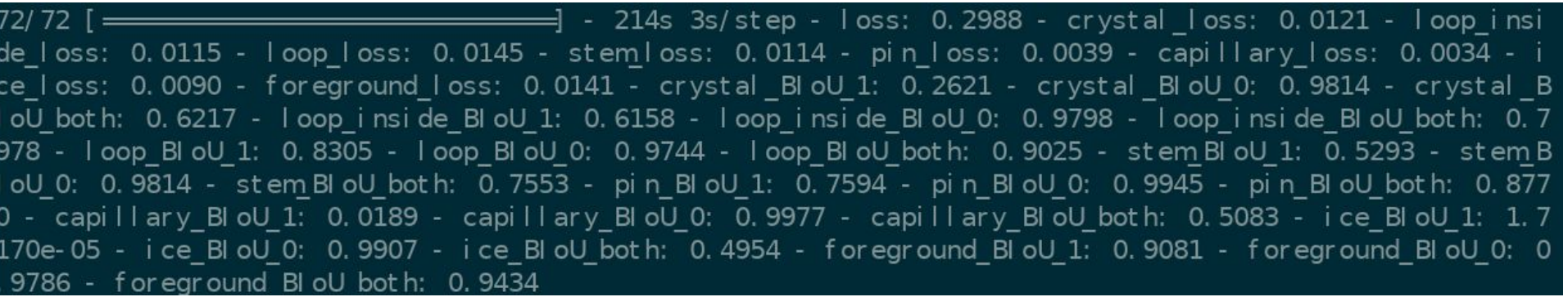
Current state-of-the-art object detectors are based on a two-stage, proposal-driven mechanism. As popularized in the R-CNN framework [11], the first stage generates a *sparse* set of candidate object locations and the second stage classifies each candidate location as one of the foreground classes or as background using a convolutional neural network. Through a sequence of advances [10, 28, 20, 14], this two-stage framework consistently achieves top accuracy on the challenging COCO benchmark [21].

Despite the success of two-stage detectors, a natural question to ask is: could a simple one-stage detector achieve similar accuracy? One stage detectors are applied over a regular, *dense* sampling of object locations, scales, and aspect ratios. Recent work on one-stage detectors, such as YOLO [26, 27] and SSD [22, 9], demonstrates promising results, yielding faster detectors with accuracy within 10-40% relative to state-of-the-art two-stage methods.

This paper pushes the envelop further: we present a one-stage object detector that, for the first time, matches the state-of-the-art COCO AP of more complex two-stage de-

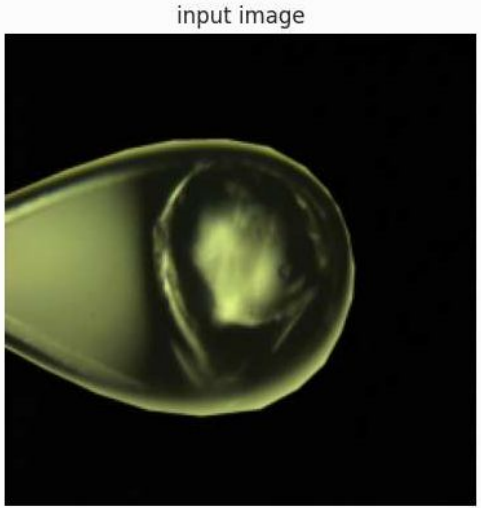
Loss function and Metrics

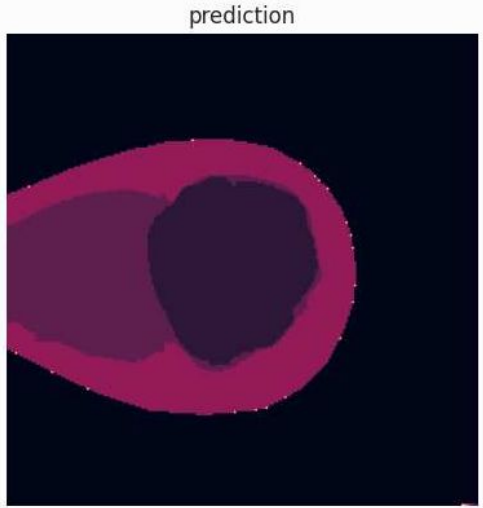
- Evaluation on all data

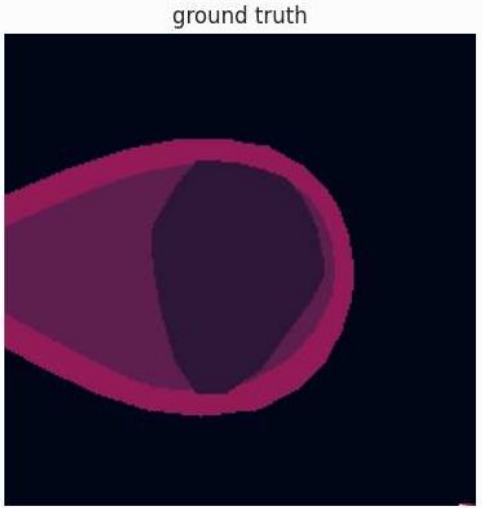


Performance

- 12ms per image in batch mode on NVIDIA RTX 6000
  - will be slower on CPU but still near real time
- robust with respect to orientation and scale

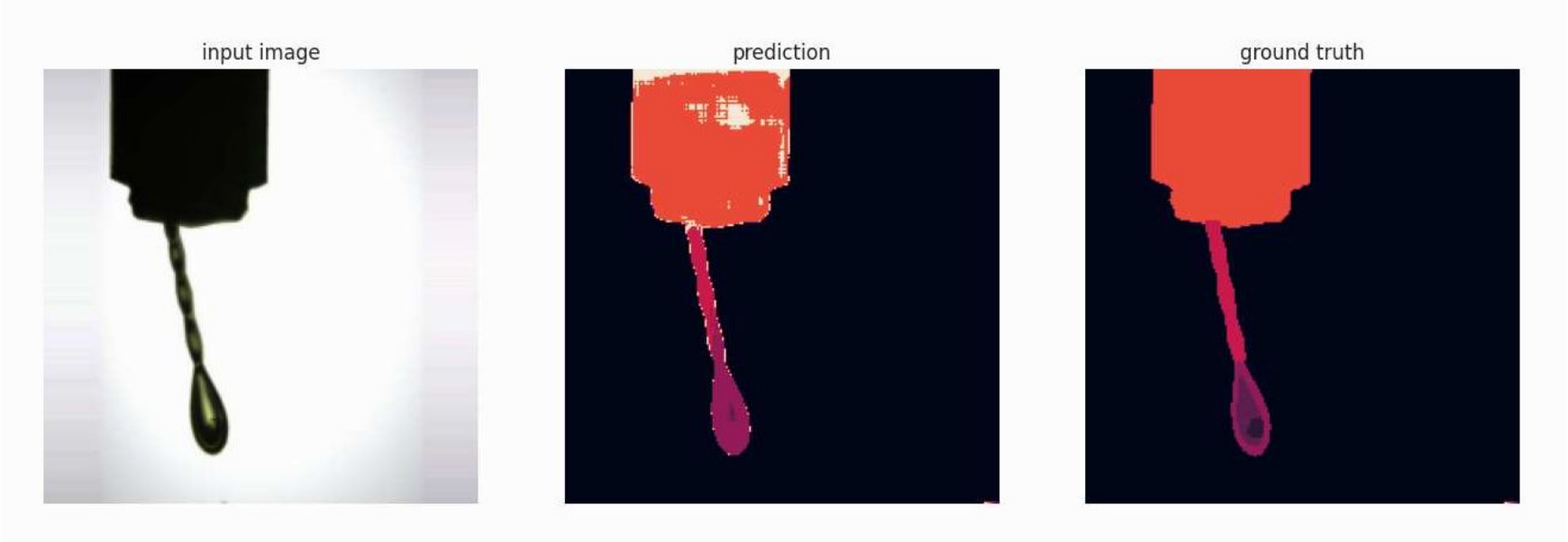






Performance

- 12ms per image in batch mode on NVIDIA RTX 6000
  - will be slower on CPU but still near real time
- robust with respect to orientation and scale



Outlook

- Bigger and better dataset
  - Please, please send me your backgrounds
- Deploying on PX2 and PX1
- Deploying at other sites
- Use for better alignment of samples
  - more accurate models of sample image movement as function of orientation require more than just three clicks
- Use for automated and accurate sample reorientation for multi orientation experiments