**Current Approach**

|  |
| --- |
| **Step 1: Barcode Region Localization**  Split image into a grid layout and classify each square patch as either part of a barcode (Class-1) or background (Class-0). See below for details on how the classifier was trained. The images below show the classifiers output. Red squares are detected as Class-1 (barcode region) and green squares are Class-0 (background).  Three different patch sizes (64, 80, 128) were tried; results were similar for patch sizes 64 and 80, which were better than the results for patch size 128  Possible alternative approaches for this step could be to use a RetinaNet like object detection model or Hough voting based approaches. These are discussed below in the Design Choices section. |
| |  |  |  | | --- | --- | --- | | Chart  Description automatically generated | Chart  Description automatically generated | Chart  Description automatically generated | |
| **Step 2: Barcode Region Expansion**  2a. A binary mask is created using Class-1 patches  2b. Morphological dilation is performed to connected patches  2c. Connected Components Analysis is performed to detect candidate barcode regions  2d.Filter regions based on area, to keep only those with area > 2500 |
| |  |  |  | | --- | --- | --- | | Chart  Description automatically generated with medium confidence | Qr code  Description automatically generated | Chart, histogram  Description automatically generated | | Chart  Description automatically generated | Chart, waterfall chart  Description automatically generated | Chart  Description automatically generated | |
|  |
| **Step 3: Barcode Region Extraction and Transformation**  **3a.** For each candidate region compute the a tight rotated bounding box  **3b.** Rotate barcode region and bounding box so that bars are vertical and bounding box is axis aligned  **3c.** Crop axis aligned bounding box region and decode data |
| |  |  |  | | --- | --- | --- | | Chart  Description automatically generated | A picture containing text  Description automatically generated | Chart  Description automatically generated | |  |  |  | |

**Supervised Training**

* The classifier was trained in a supervised manner
* 33 randomly selected images from both datasets (with AF and without AF) were manually labelled
* These were split in 22 training and 11 validation images
* For each training image 9 augmented (randomly transformed) versions were added, resulting in a total training set of 220 (22 x (1 original + 9 augmented)) images. Data augmentation was implemented using the following repo: <https://github.com/Paperspace/DataAugmentationForObjectDetection>. Each augmented image undergoes a random combination of rotation, translation, scaling, sheering, horizontal flip and colour re-mapping in the HSV space.
* Each training image is then split into a grid layout as shown above and squares that have at least 50% of their area inside a human-labelled barcode region, are assigned label 1.
* During each training step, balanced batches of 40 patches (20 of class 1 and 20 of class 0) were sampled from 4 images. That is, five class 1 and five class 0 patches were extracted from each image. Images were shuffled after each epoch. Each image participated once in an epoch and therefore in 200 epochs, ~2000 patches are sampled from each image.
* Relatively simple models with approximately 200,000 to 350,000 trainable params (depending on patch size) were used. The architecture for patch size 80 x 80 is shown below
* The model was trained for 200 epochs and training and validation accuracies were as shown below. The model achieved ~85% training and validation accuracies.

|  |  |
| --- | --- |
| Model Architecture | Training Performance |
| Table  Description automatically generated | Chart  Description automatically generated |

**Performance**

Patch Size = 80 x 80

Dataset1 (with AF) barcode extraction accuracy is **0.33**

Dataset2 (without AF) barcode extraction accuracy is **0.069**

Patch Size = 64 x 64

Dataset1 (with AF) barcode extraction accuracy is **0.353**

Dataset2 (without AF) barcode extraction accuracy is **0.10**

Here barcode extraction accuracy means…

Few false negatives, more false positives

**Design choices**

1. Alternative choices for supervised barcode region detection
   1. RetinaNet
2. Barcode region detection using CV methods only
   1. Hough transform
   2. Speed considerations
3. Sharpening blurred barcode using deep learning
   1. There is also the possibility of training a deblurring autoencoder in an unsupervised manner
4. Decoding using deep learning

**Challenges / Improvements**

1. Misclassification with text regions

|  |  |  |
| --- | --- | --- |
| Chart  Description automatically generated | Chart  Description automatically generated | Chart  Description automatically generated |
|  |  |  |

1. Low Coverage
2. Non-linear distortions
3. Out-of-focus images
4. Accounting for Scale
   1. In the current implementation only a single, heuristically optimised, grid resolution is used. The grid resolution however should be based on image scale. The figure below shows two images with very different scales. Moreover, at higher grid resolutions, each square does not capture sufficient context. In a very local region, barcode and text can appear similar. Therefore, it can be useful to consider a hierarchy of grid resolutions and judge whether a region contains barcode based on the consensus across the hierarchy.

|  |  |
| --- | --- |
| Chart  Description automatically generated | Chart  Description automatically generated |
|  |  |

1. The classifier still confuses between text and barcode
   1. Labelling both barcode and text regions
2. Improving coverage by iteratively growing detection
   1. Example of low coverage

|  |
| --- |
| A picture containing chart  Description automatically generated |
|  |

1. Sampling strips from barcode region to get multiple decodings