Mosaic-SR: An Adaptive Multi-step Super-Resolution Method for Low-Resolution 2D Barcodes

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

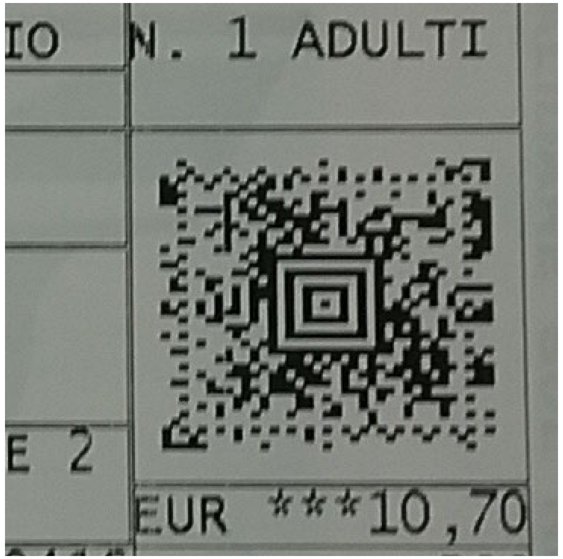
DATALOGIC

ICIP 2025, ANCHORAGE

Enrico Vezzali, L. Vorabbi, C. Grana, F. Bolelli

2D Barcodes

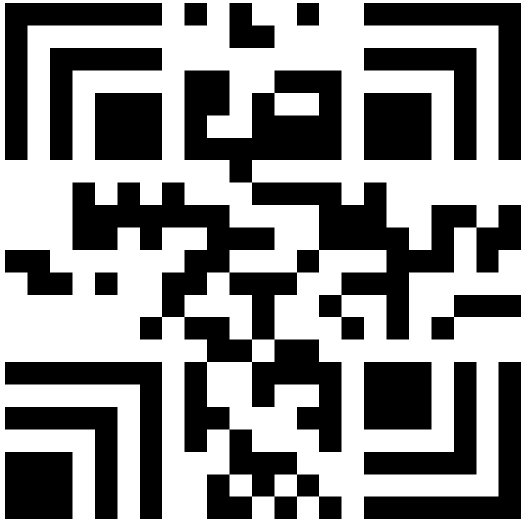
- 2D barcodes store data both horizontally and vertically using patterns like squares or dots
- The widely known QR Code was invented in 1994 by Denso Wave in Japan
- The main advantage is **Higher Data Capacity** compared to linear barcodes



Aztec Code

QR Code

Datamatrix



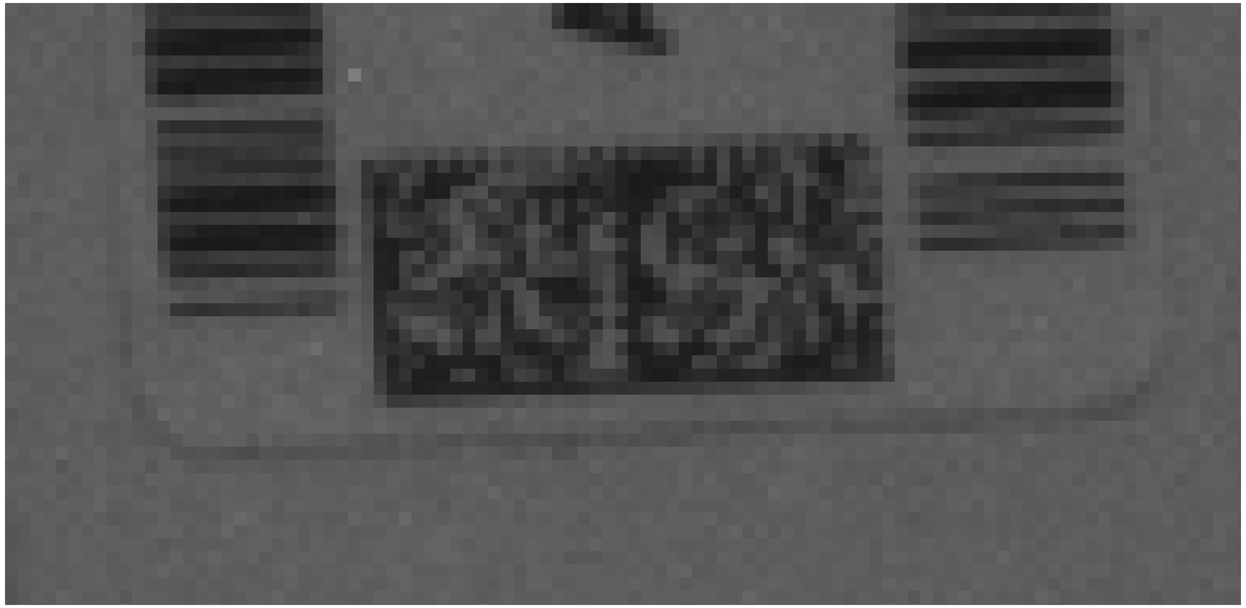
Han Xin Code

2D Barcodes – Range Issues

- In many applications, it is necessary to read 2D barcodes from a distance
- For example, in component tracking in industrial pipelines, depending on the size of the objects
- In warehouses, where there could be parcels on very high shelves

2D Barcodes – Range Issues

- This often results in images with very low pixel density, making them difficult to read
- If the resolution is too low, the critical distinction between black and white modules is lost



Defining the Super-Resolution Task for 2D Barcodes

- We address the problem of $2 \times$ super-resolution,
- Each pixel $I(x, y)$ in a low-resolution image I corresponds to a 2×2-block in the high-resolution image I_{HD}.

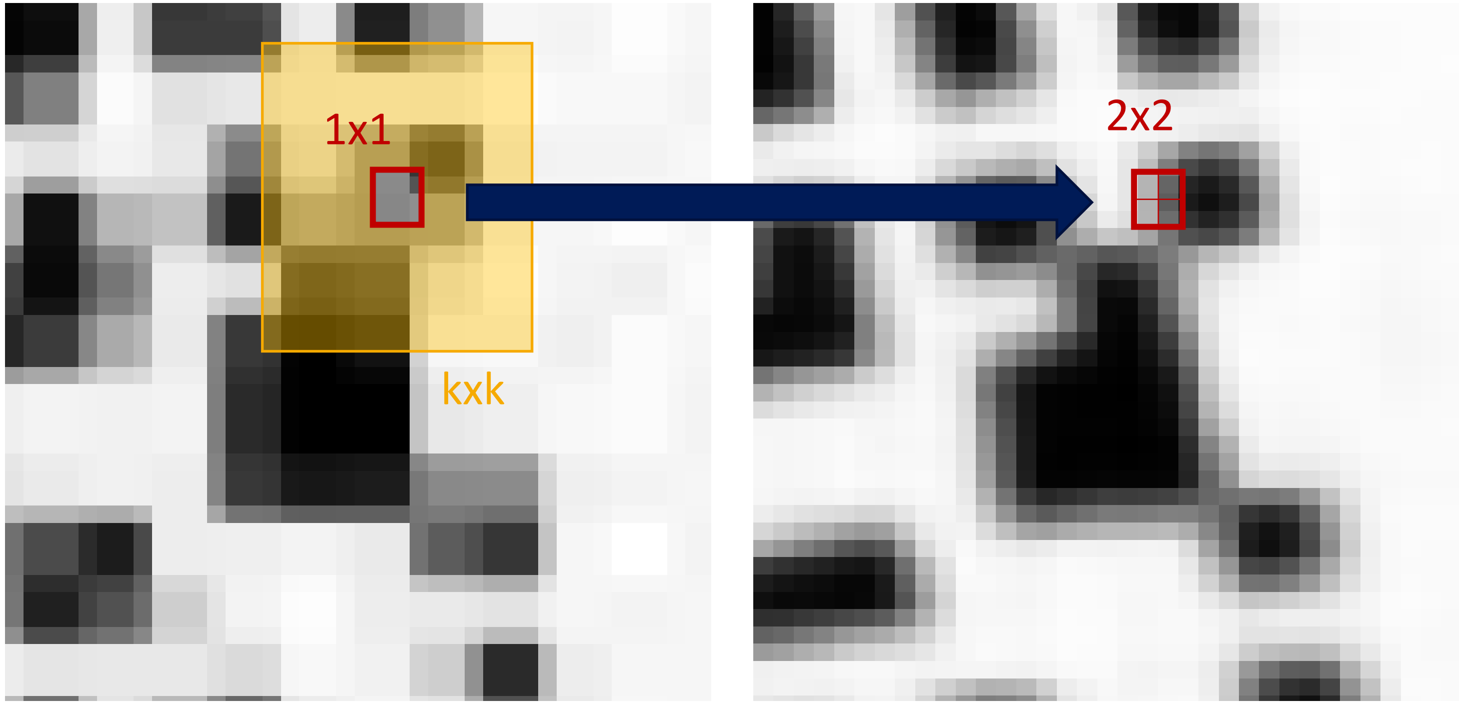
We seek a function f_θ that, given a local patch around $I(x, y)$, of size $k \times k$, approximates the corresponding 2×2 block in I_{HD}

$$\boldsymbol{f_\theta}: \mathbb{R}^{\boldsymbol{k \times k}} \rightarrow \mathbb{R}^{\boldsymbol{2 \times 2}}$$

We define f_θ as space-invariant

Defining the Super-Resolution Task for 2D Barcodes

- We address the problem of $2 \times$ super-resolution,
- Each pixel $I(x, y)$ in a low-resolution image I corresponds to a 2×2-block in the high-resolution image I_{HD}.



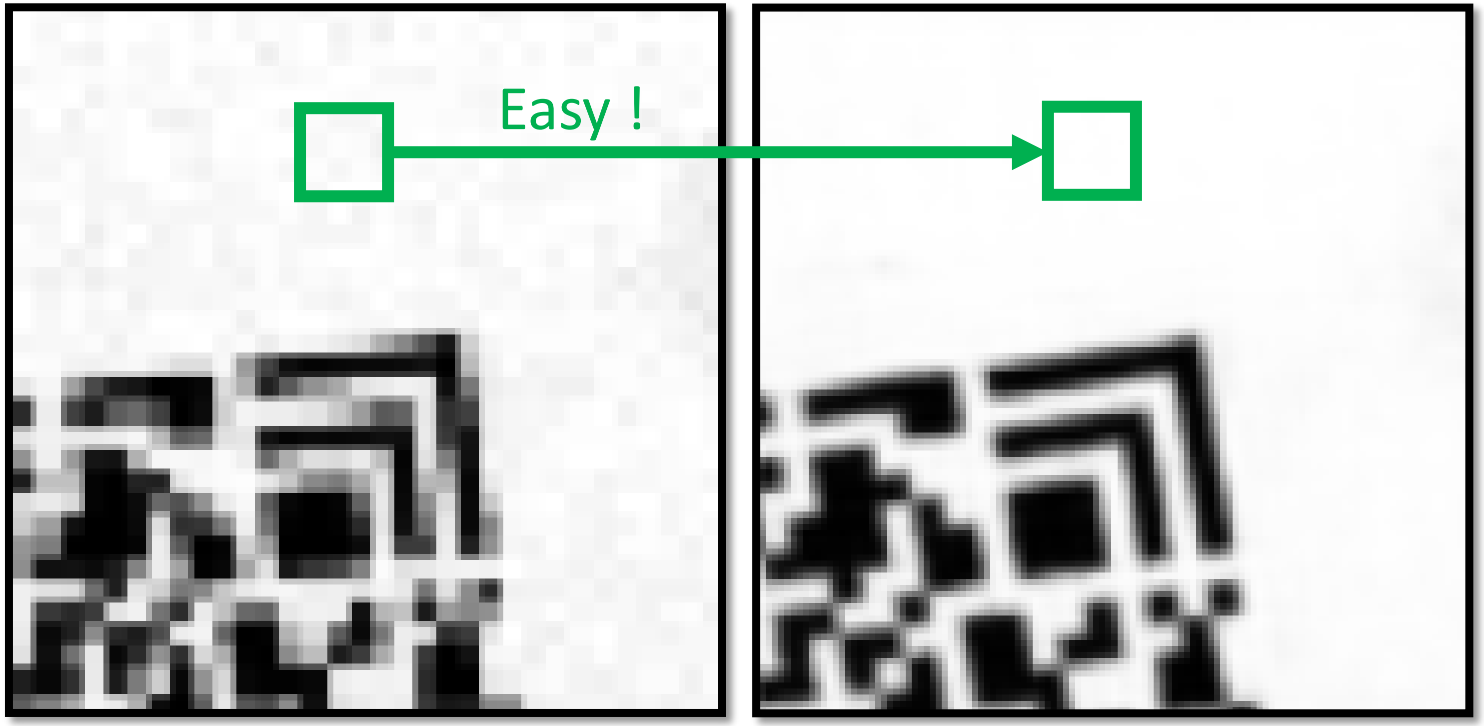
We seek a function f_θ that, given a local patch around $I(x, y)$, of size $k \times k$, best approximates the corresponding 2×2 block in I_{HD}

$$\boldsymbol{f_\theta}: \mathbb{R}^{\boldsymbol{k \times k}} \rightarrow \mathbb{R}^{\boldsymbol{2 \times 2}}$$

We define f_θ as space-invariant

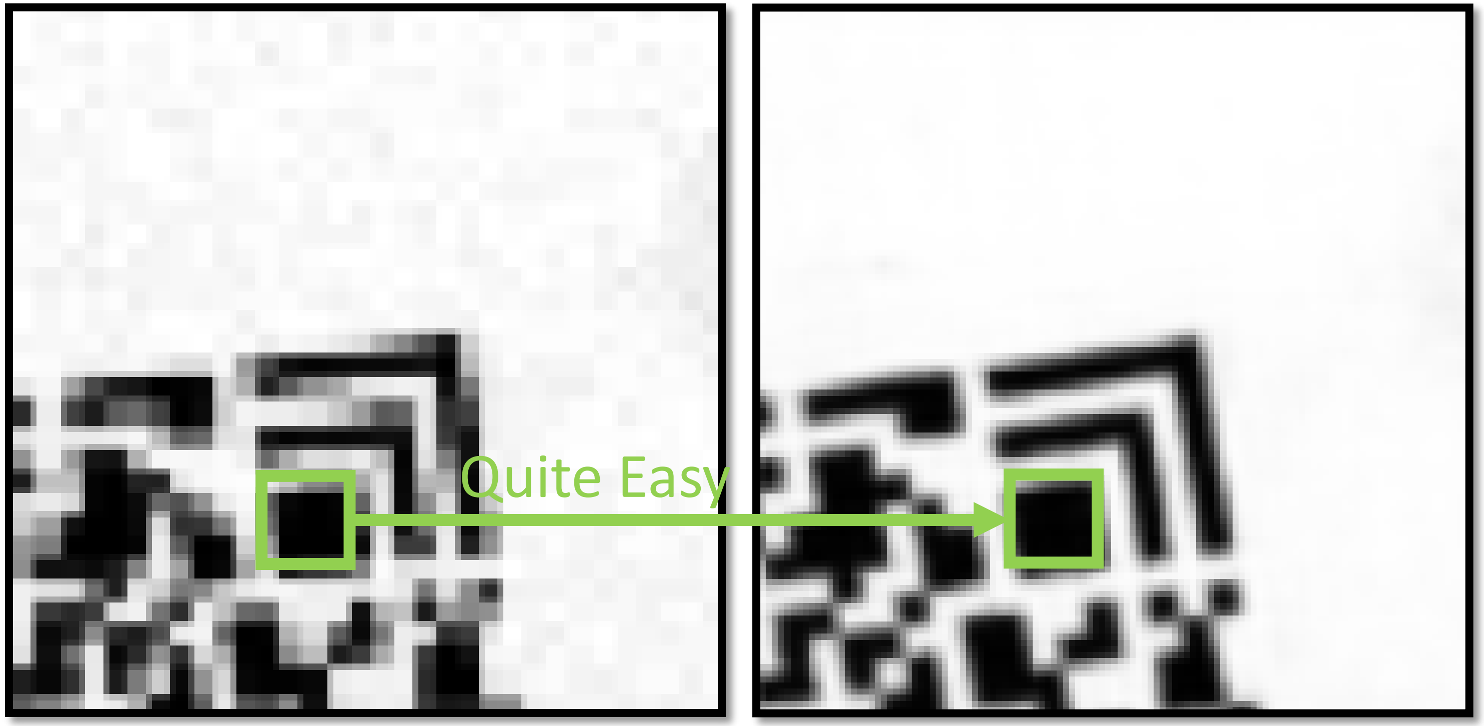
Our Proposal – Mosaic-SR Intuition

- Not all areas of the image are equally difficult to upscale
- Upscaling background or uniform areas is **straightforward**
- Upscaling **sharp edges and intricate corners** – critical for barcode readability – is the most challenging part
- It would be great to **limit the number of computations in the areas that do not require it**



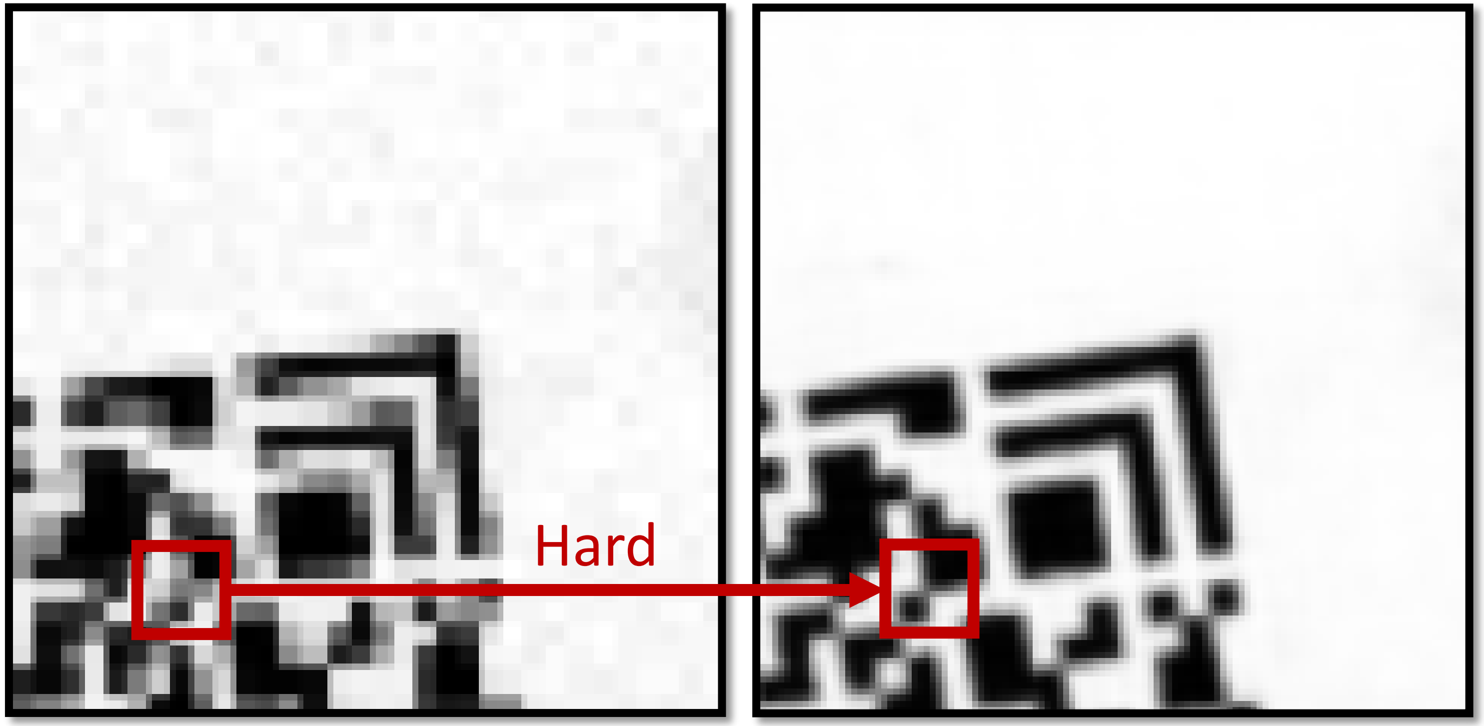
Our Proposal – Mosaic-SR Intuition

- Not all areas of the image are equally difficult to upscale
- Upscaling background or uniform areas is **straightforward**
- Upscaling **sharp edges and intricate corners** – critical for barcode readability – is the most challenging part
- It would be great to **limit the number of computations in the areas that do not require it**



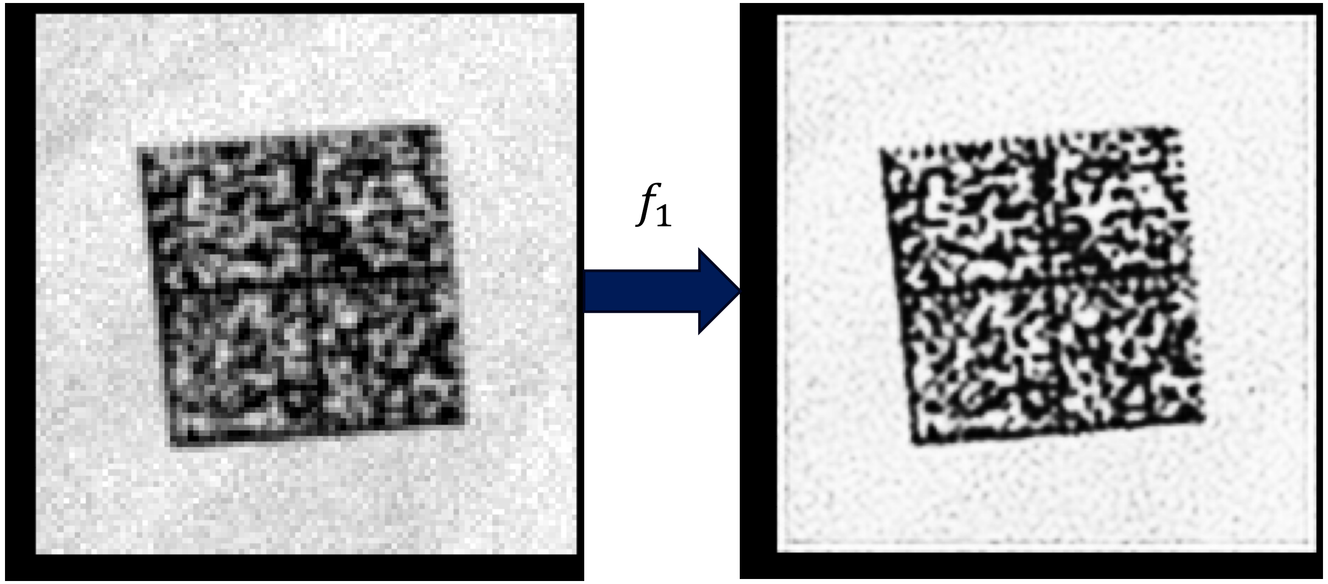
Our Proposal – Mosaic-SR Intuition

- Not all areas of the image are equally difficult to upscale
- Upscaling background or uniform areas is **straightforward**
- Upscaling **sharp edges and intricate corners** – critical for barcode readability – is the most challenging part
- It would be great to **limit the number of computations in the areas that do not require it**



Our Proposal – Mosaic-SR Intuition

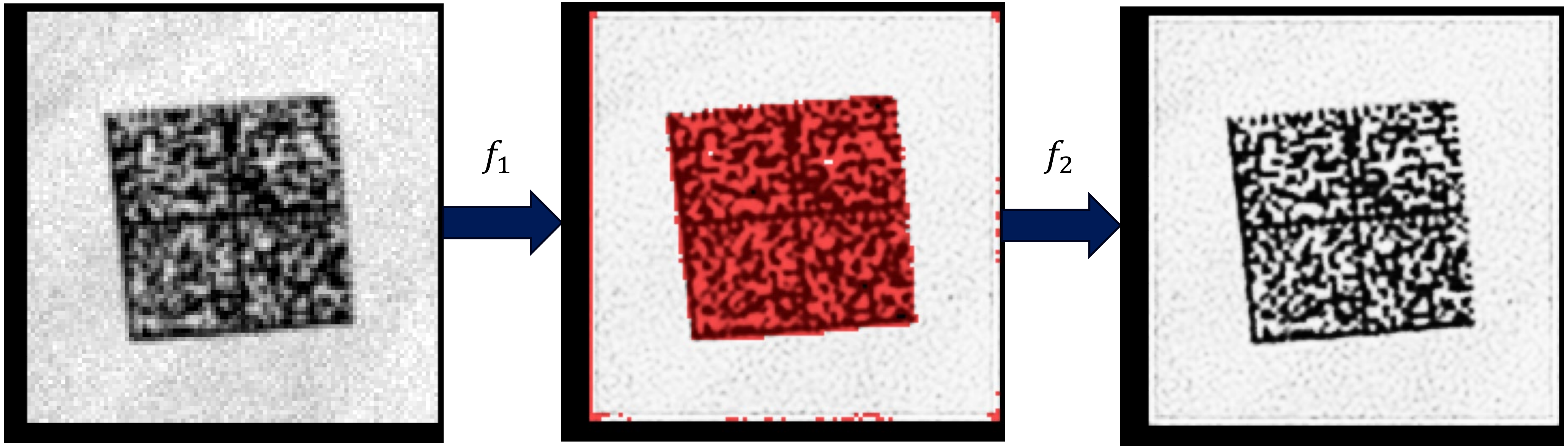
- The idea of Mosaic-SR is to upscale the image iteratively
- At each step, just the areas that need further refinement are processed



A fast-to-compute function f_1 is used to upscale all patches

Our Proposal – Mosaic-SR Intuition

- The idea of Mosaic-SR is to upscale the image iteratively
- At each step, just the areas that need further refinement are processed

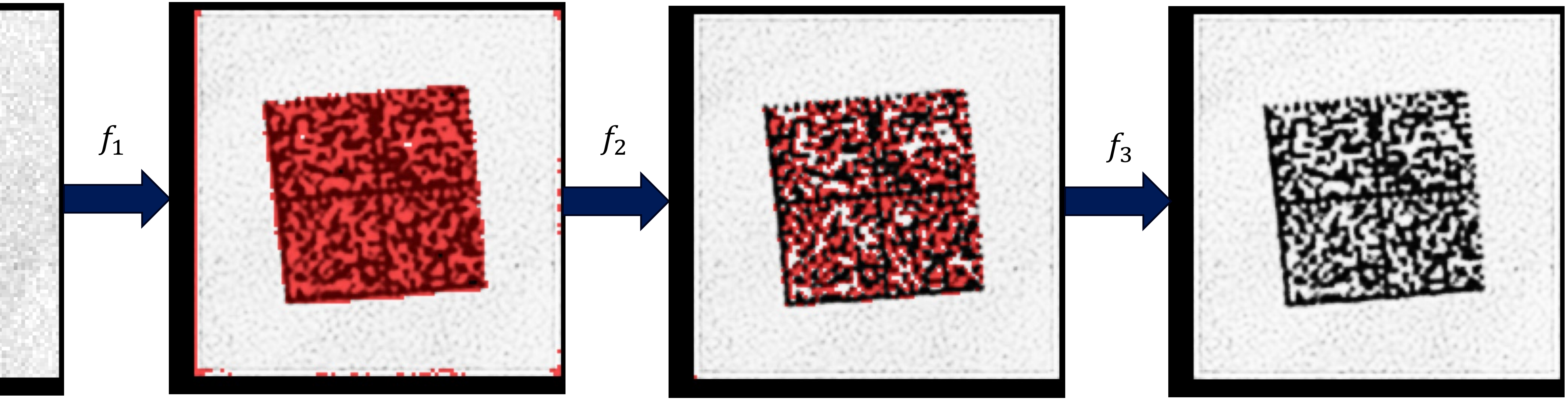


A fast-to-compute function f_1 is used to upscale all patches

The areas that need further improvements are enhanced with function f_2

Our Proposal – Mosaic-SR Intuition

- The idea of Mosaic-SR is to upscale the image iteratively
- At each step, just the areas that need further refinement are processed



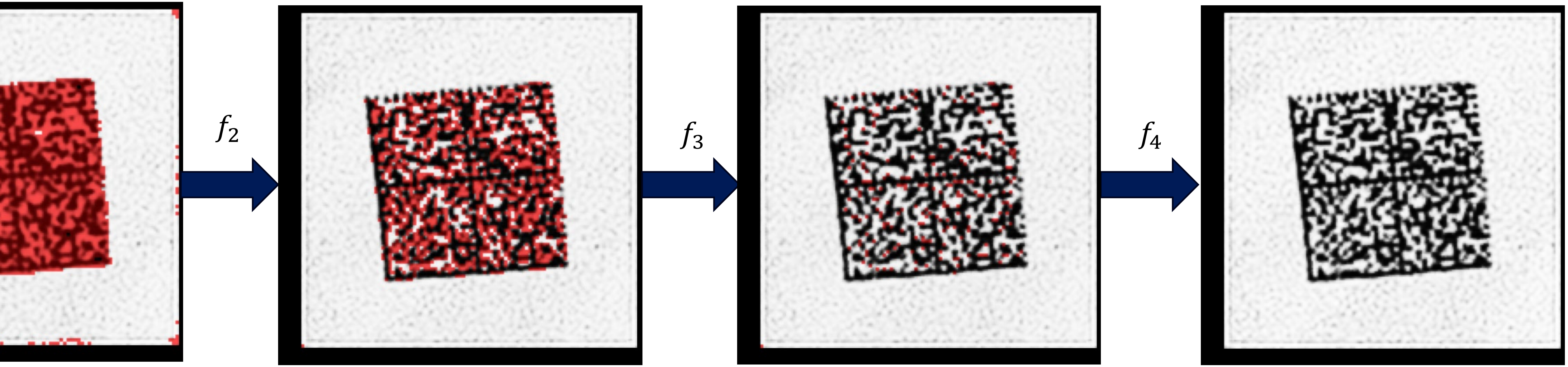
compute function f_1 is
upscale all patches

The areas that need further improvements are enhanced with function f_2

The areas that need further improvements are enhanced with function f_3

Our Proposal – Mosaic-SR Intuition

- The idea of Mosaic-SR is to upscale the image iteratively
- At each step, just the areas that need further refinement are processed



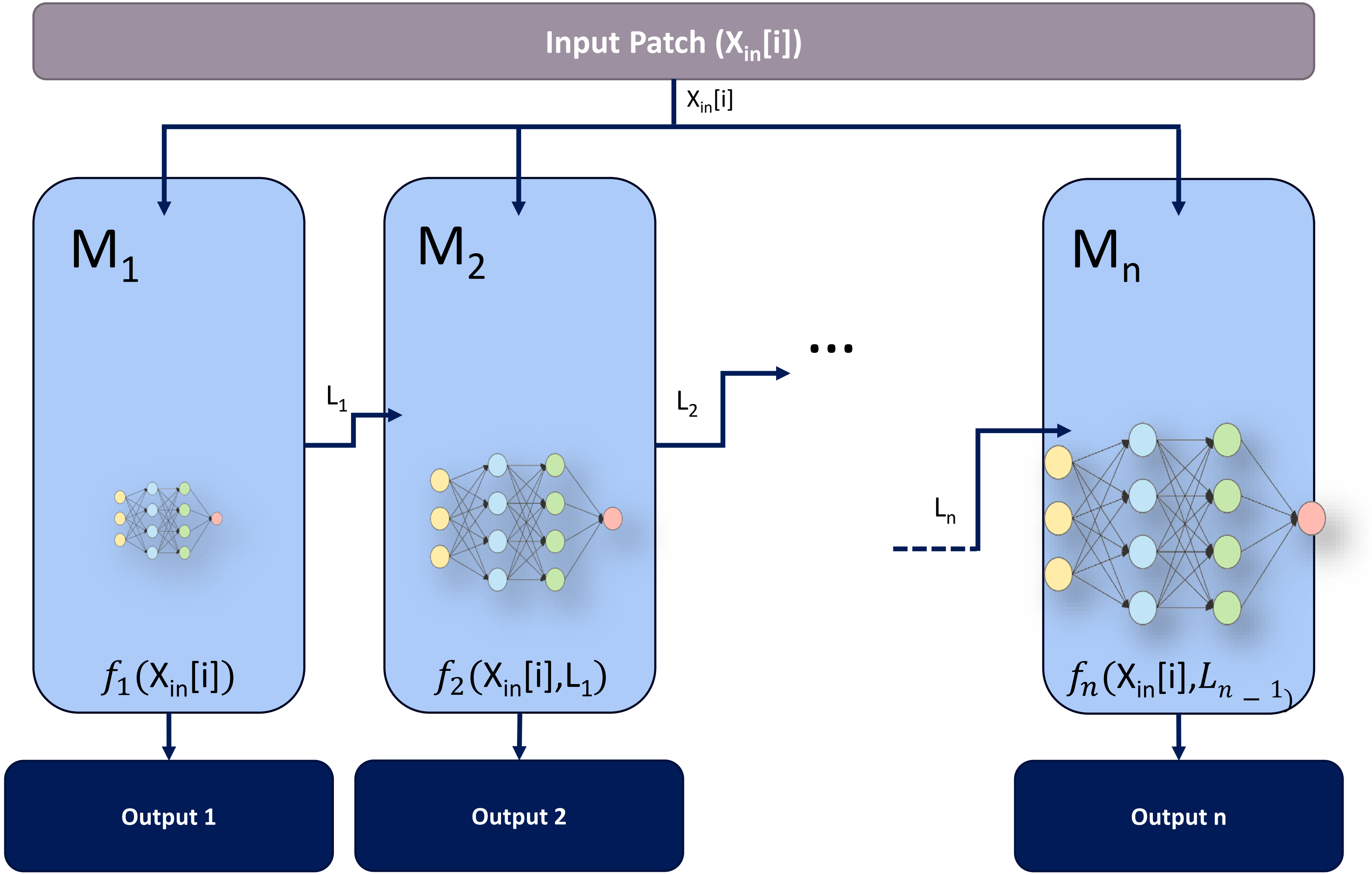
The areas that need further improvements are enhanced with function f_2

The areas that need further improvements are enhanced with function f_3

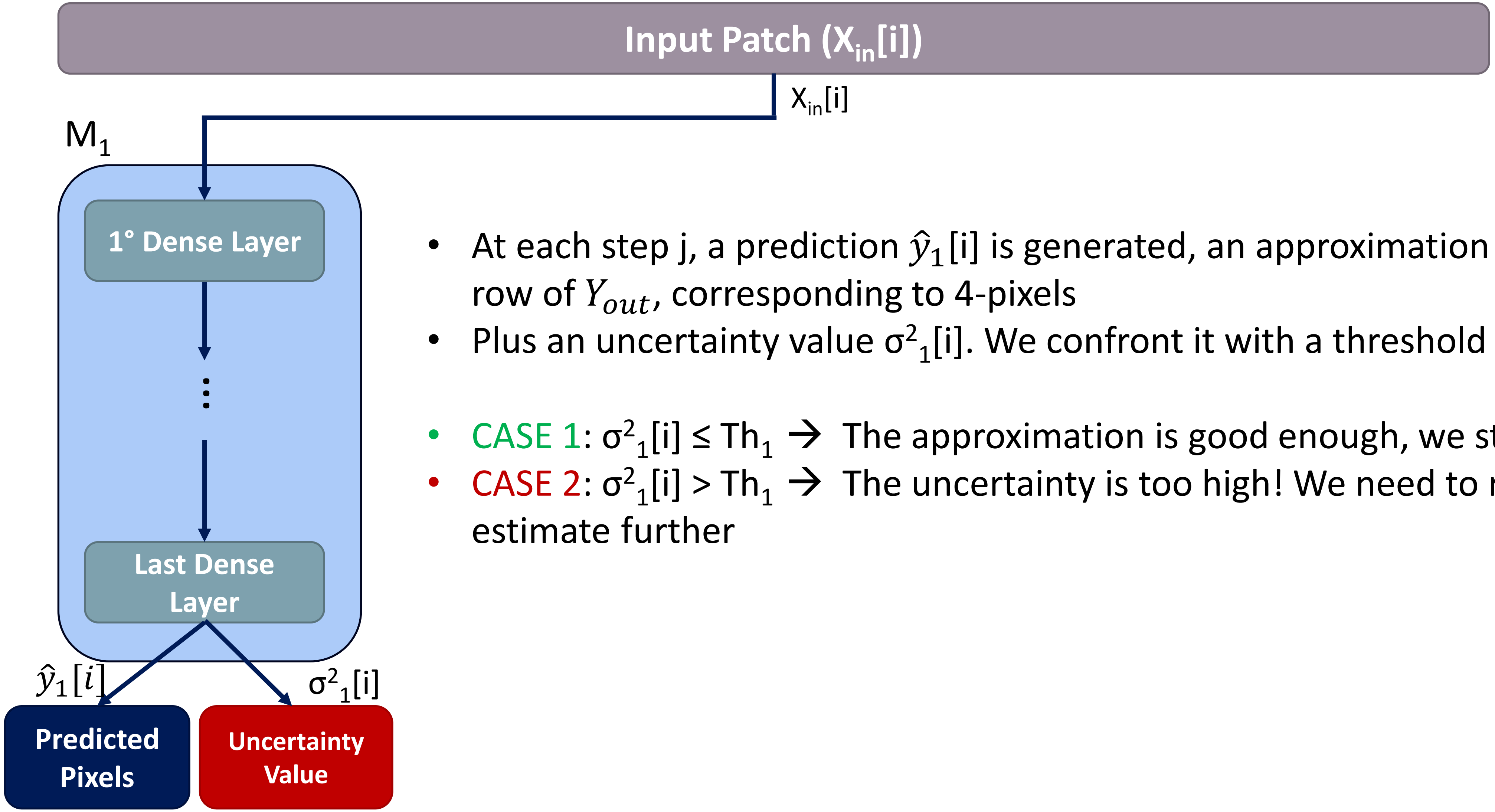
And so on

Mosaic-SR Architecture

- Function f_i is performed by the neural network $M_i, i = 1, ..n$
- We want M_i to be faster than M_{i+1}
- Each function takes as input $X_{in}[i]$ and an internal result (L) from the previous network



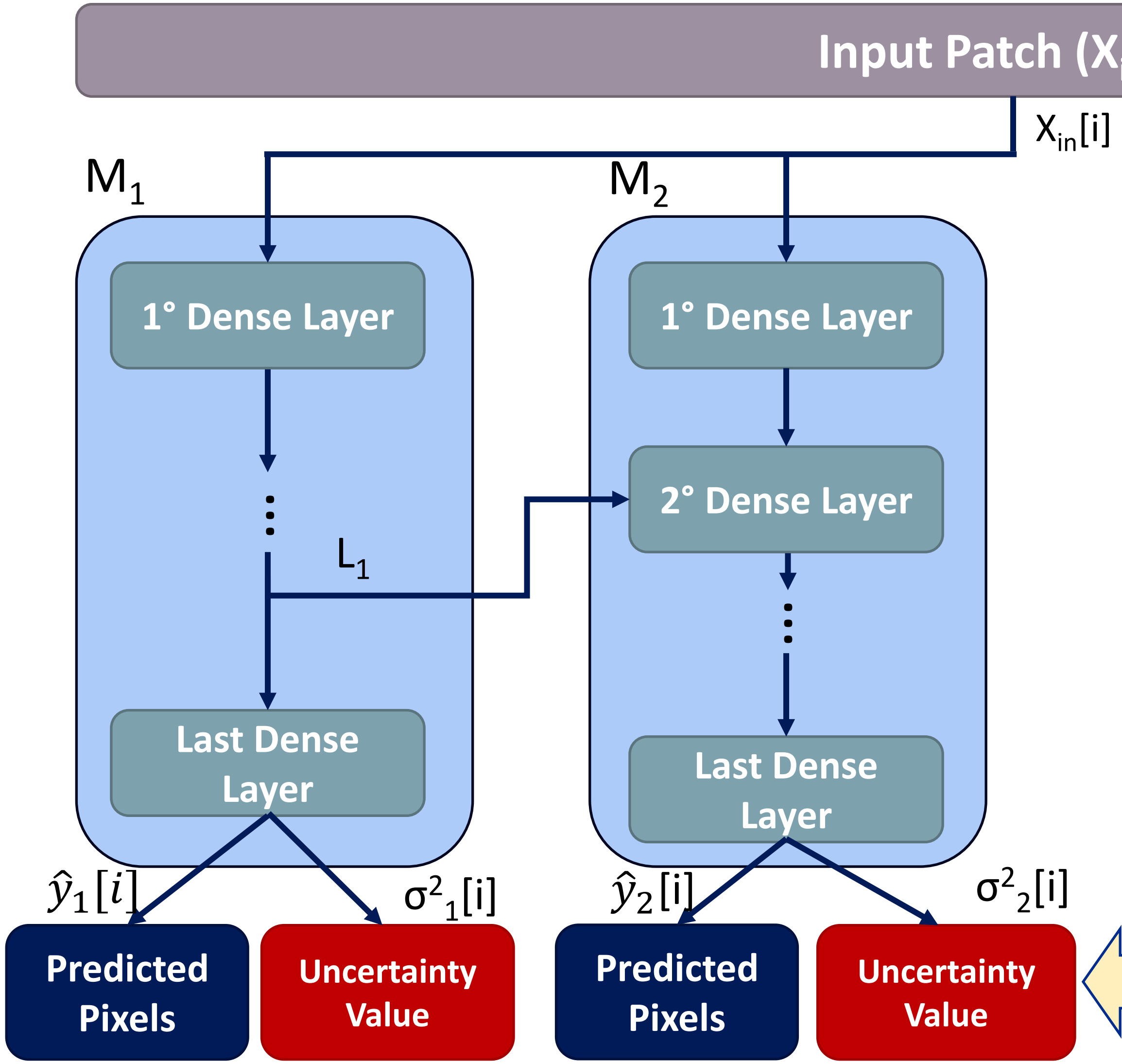
Mosaic-SR Architecture – Detailed



- At each step j, a prediction $\hat{y}_1[i]$ is generated, an approximation of the i-th row of Y_{out}, corresponding to 4-pixels
- Plus an uncertainty value $\sigma^2_1[i]$. We confront it with a threshold Th_1

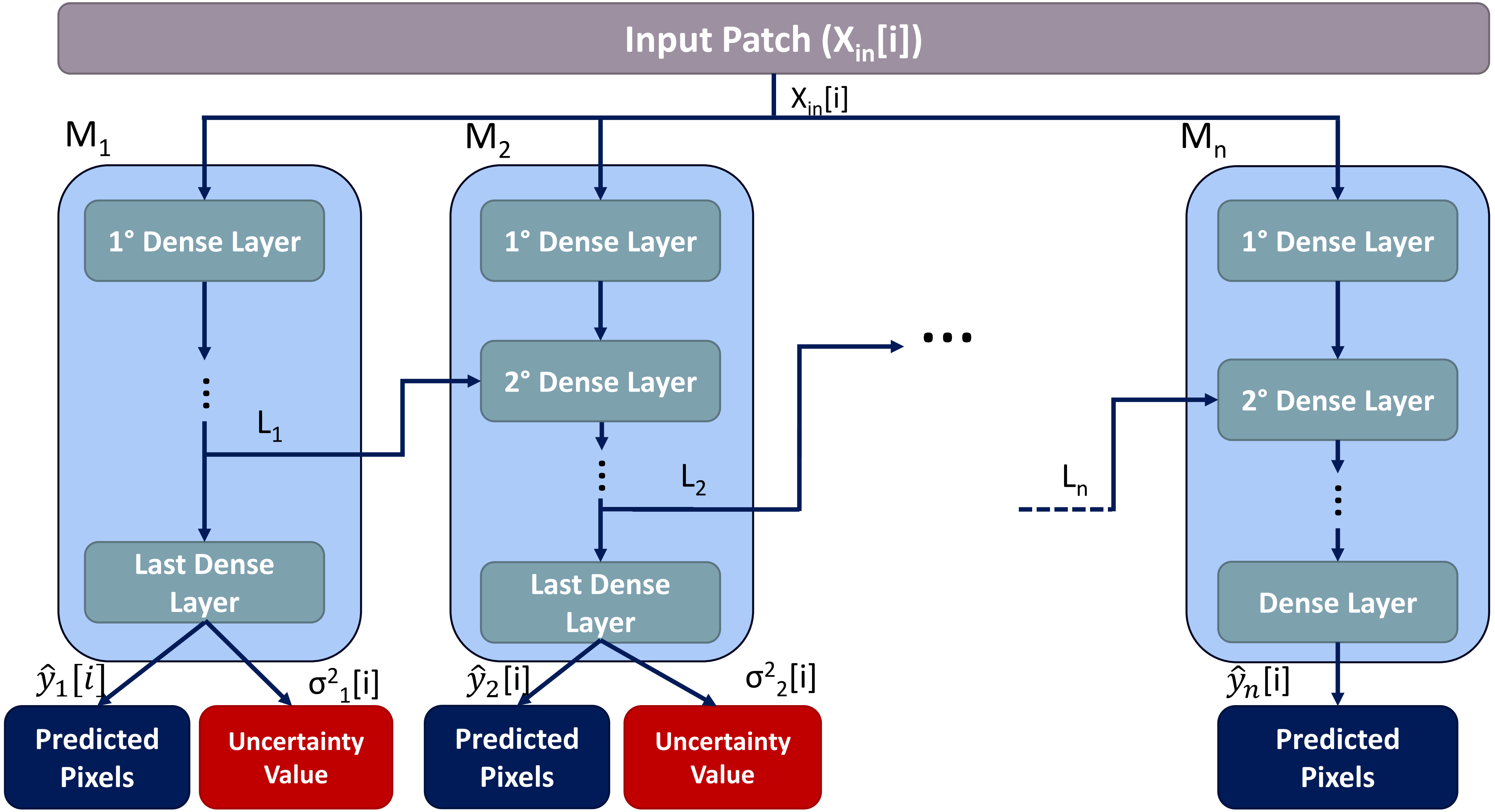
- CASE 1: $\sigma^2_1[i] \leq Th_1$ → The approximation is good enough, we stop here
- CASE 2: $\sigma^2_1[i] > Th_1$ → The uncertainty is too high! We need to refine our estimate further

Mosaic-SR Architecture – Detailed



If $\sigma^2{}_2[i]$ is lower than the threshold Th_2 we stop.
Otherwise, we continue with the next steps

Mosaic-SR Architecture – Detailed



Predicting the distribution of a Random Variable

- **Super-Resolution is Ill-Posed**: A single low-resolution (LR) patch can correspond to multiple high-resolution (HR) outputs. A deterministic prediction isn't enough.

- **Probabilistic Solution**: We model the output as a probability distribution. We assume the prediction error follows a Gaussian distribution: $\mathcal{N}(0, \sigma^2)$

- Our Goal 🎯 : For each input patch, the network must predict two things:
    1. The Mean ($\hat{y}$): The super-resolved pixel values.
    2. The Variance (σ^2): An "uncertainty value" to decide if the prediction needs more refinement.

Likelihood Maximization: Assuming Gaussian Errors

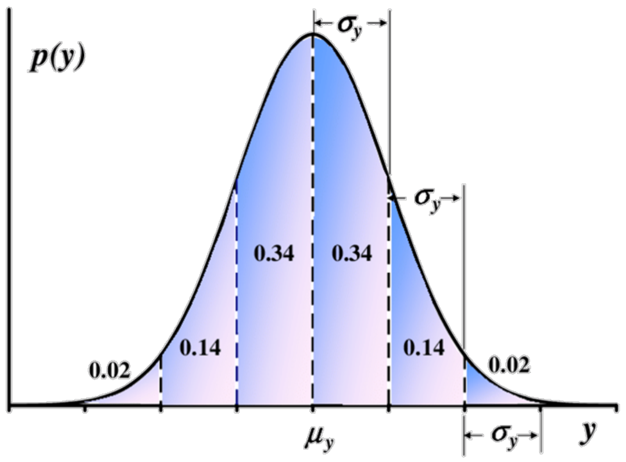
By maximizing the log-likelihood of our Gaussian assumption, we get the following cost function, for each input pixel:

$$C_i(\theta) = \frac{\|d_i - \hat{y}_i\|_2^2}{\hat{\sigma}_i^{\,2}} + 4\ln \hat{\sigma}_i^{\,2}$$

Resulting in the following loss function to train the models:

$$\mathcal{L}(\theta) = \sum_{i=0}^{N} C_i(\theta) = \sum_{i=0}^{N} \frac{\|d_i - \hat{y}_i\|_2^2}{\hat{\sigma}_i^{\,2}} + 4\ln \hat{\sigma}_i^{\,2}$$

Finally, a linear combination of this proposed loss and MSE loss makes the training more stable

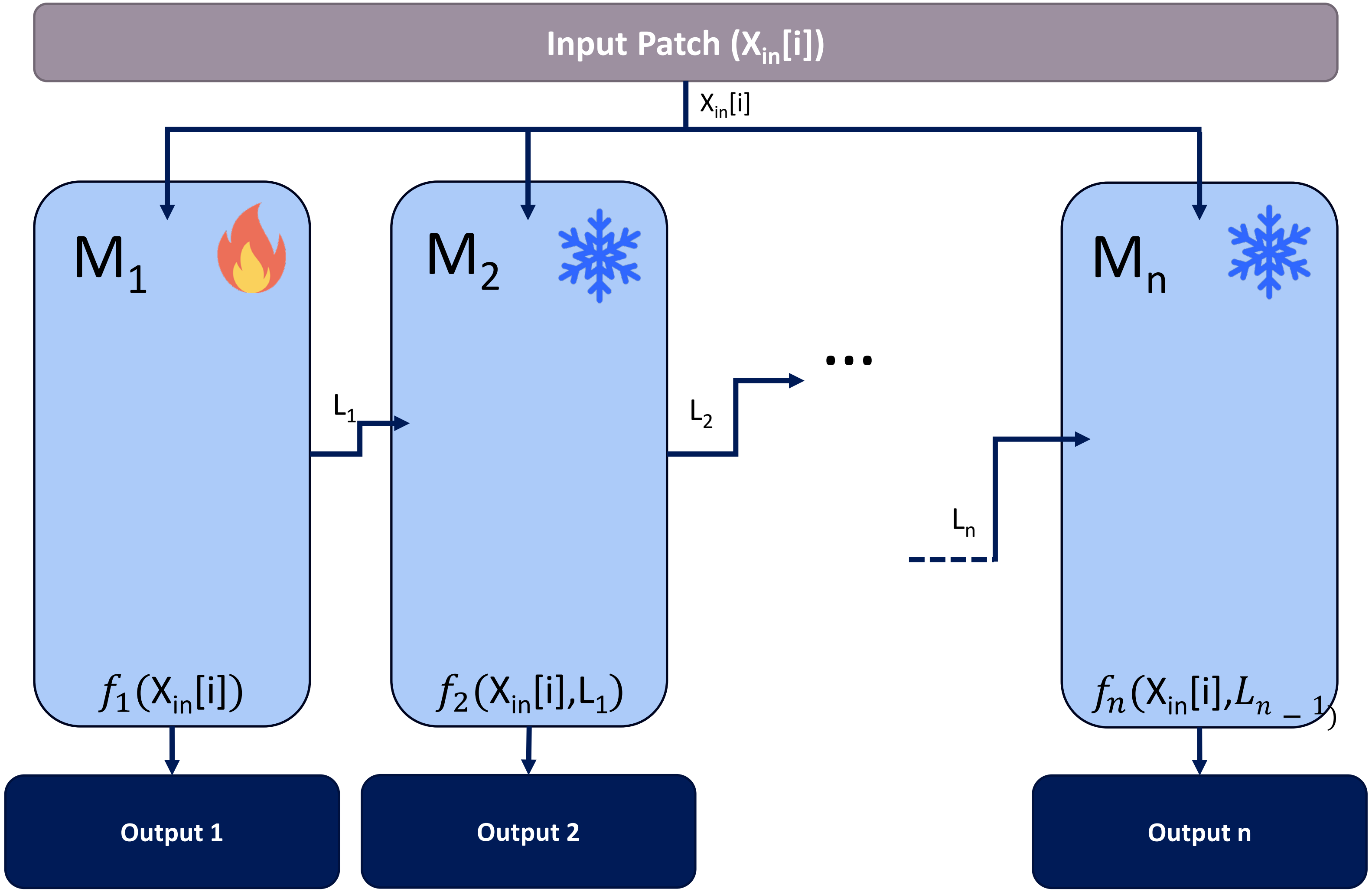


Mosaic SR: Training Strategy

Initial Training

Training all n network modules $(M_1, M_2, \ldots M_n)$ simultaneously from scratch can be problematic due to **competing gradients**, potentially leading to instability

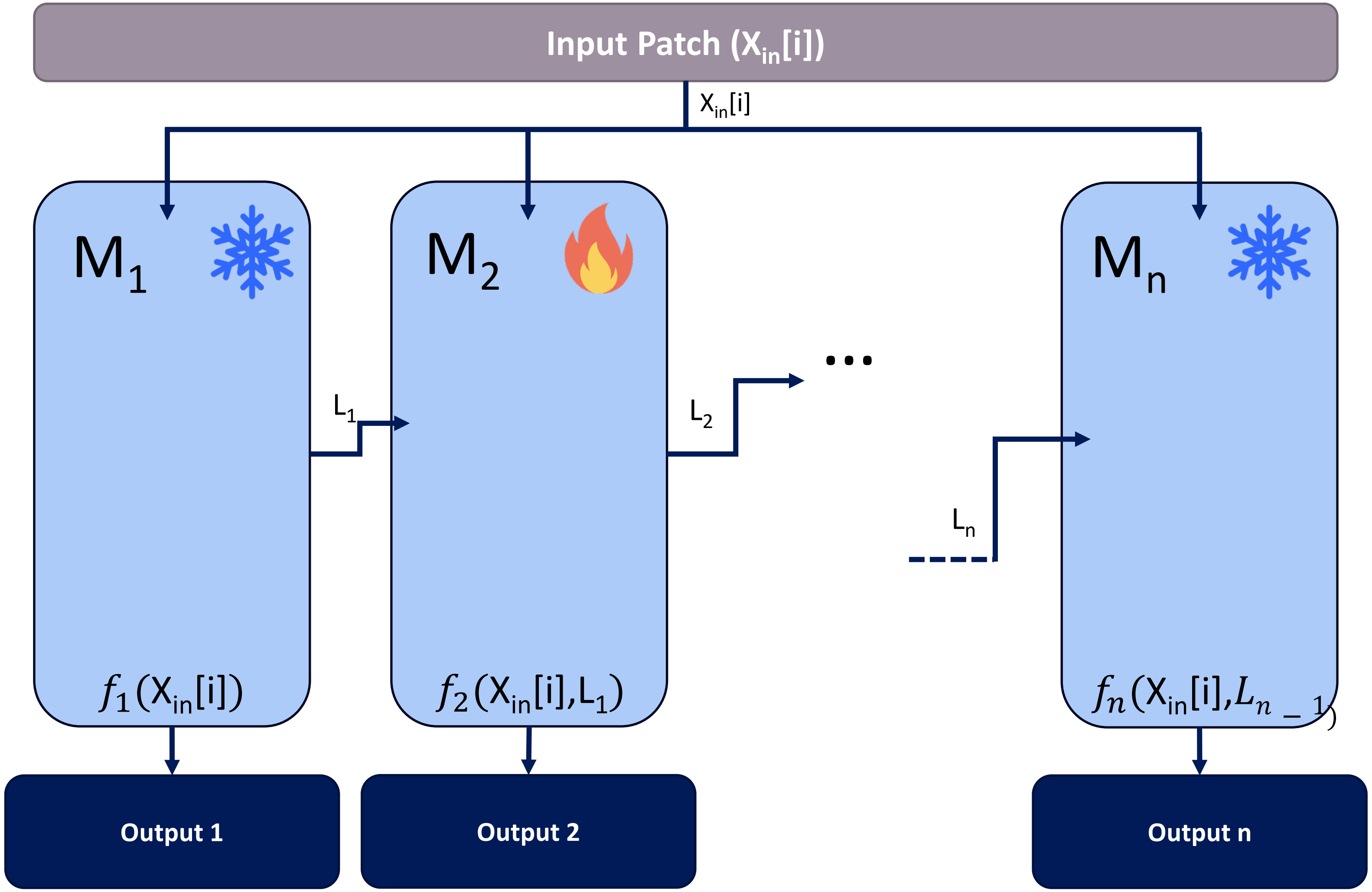
The architecture is trained one model M_k at a time, with the others frozen



Mosaic SR: Training Strategy

Initial Training
Training all n network modules $(M_1, M_2, \dots M_n)$ simultaneously from scratch can be problematic due to **competing gradients**, potentially leading to instability

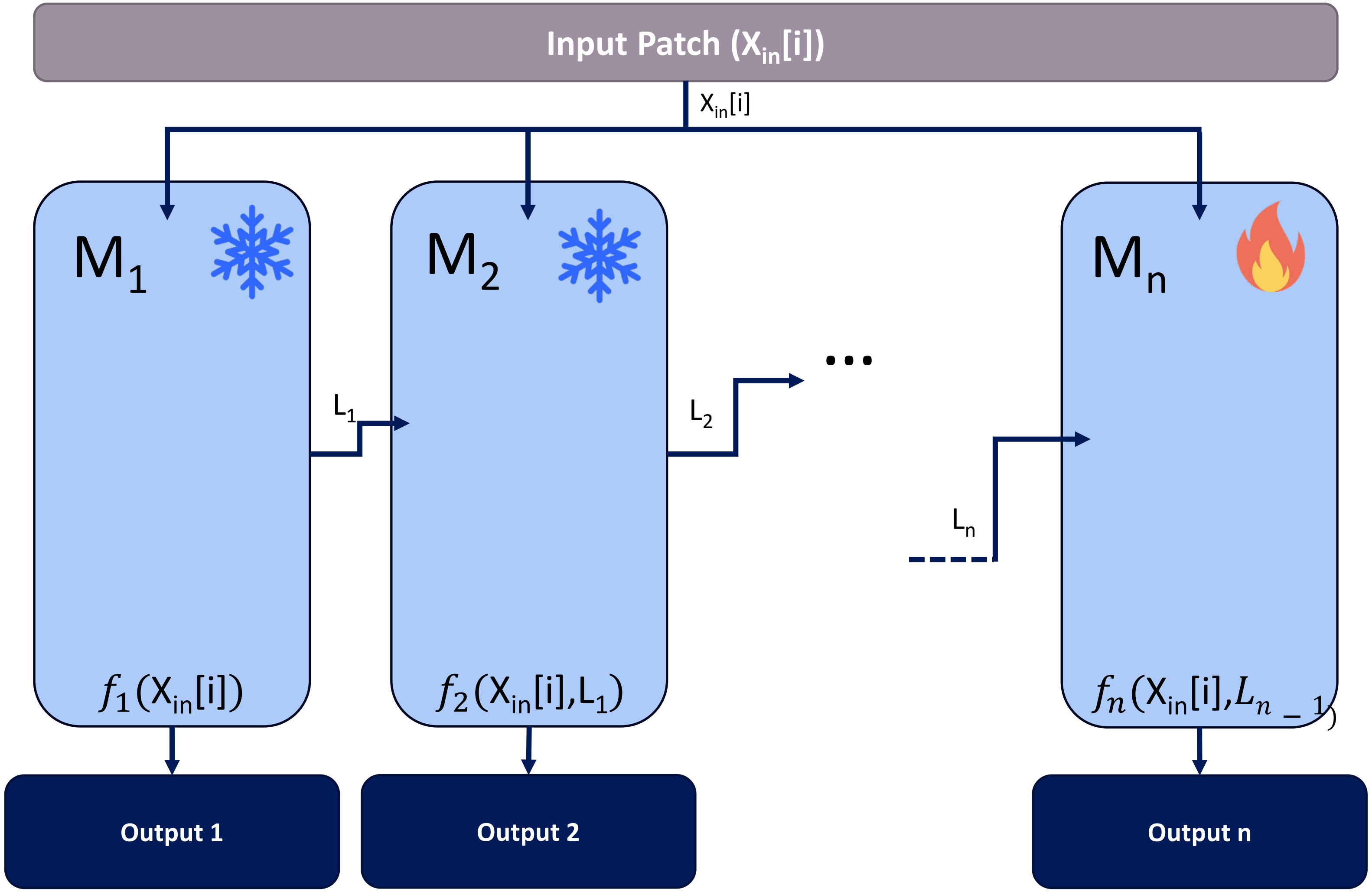
The architecture is trained one model M_k at a time, with the others frozen



Mosaic SR: Training Strategy

Initial Training
Training all n network modules $(M_1, M_2, \dots M_n)$ simultaneously from scratch can be problematic due to **competing gradients**, potentially leading to instability

The architecture is trained one model M_k at a time, with the others frozen



BarBeR Dataset

- For training we used the public dataset BarBeR, containing 8,748 barcode images (1,756 2D barcodes), captured under various conditions, such as varying lighting, noise, and obstructions
- The dataset is now publicly available
- **Includes 3 types of 2D barcodes:** QR Codes, Datamatrix and Aztec Codes

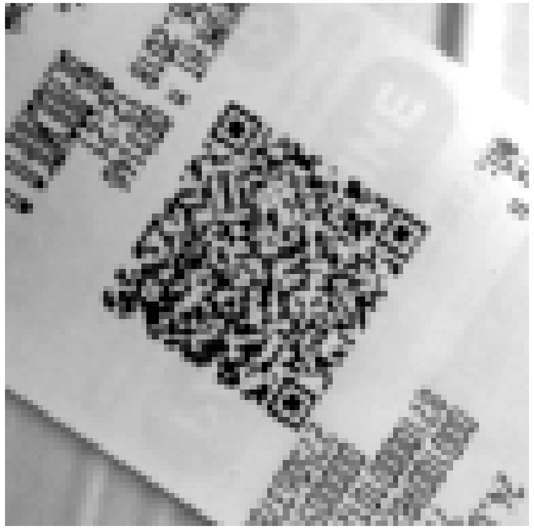
BarBeR

BarBeR Dataset

- From the dataset, we selected only 2D barcode crops with a minimum density of 3.2 PPM, producing 1,366 valid crops.
- Each crop was resized via bicubic interpolation to yield 7 low-resolution (LR) variants (1.0 PPM to 1.6 PPM, with a step of 0.1 PPM) and 7 corresponding high-resolution (HR) versions at double each LR density (2.0 PPM to 3.2 PPM)
- **This process generated 9,562 LR/HR pairs**
- All images are 128×128 pixels and converted to grayscale

1.4 PPM LD

2.8 PPM HD



1.1 PPM LD

2.2 PPM HD

Tested Architecture

For our tests we used an architecture with **three** distinct upscaling network modules: **M1,M2,** and **M3**

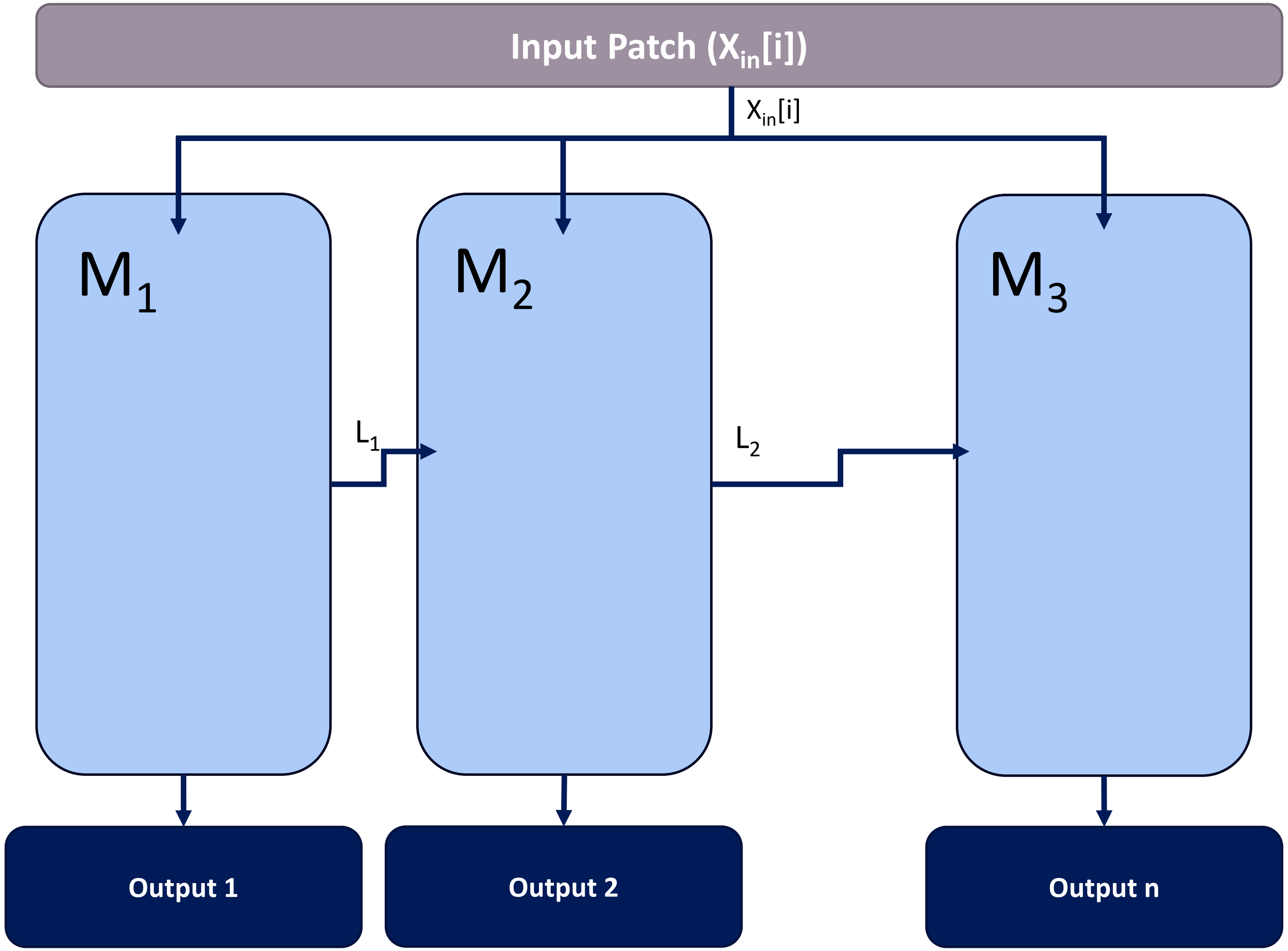
Module Specifications:

$\boldsymbol{M_1}$: 3 layers, 1,154 parameters

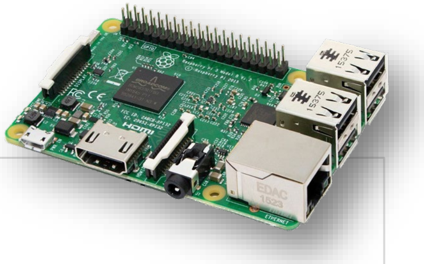
$\boldsymbol{M_2}$: 4 layers, 3,778 parameters

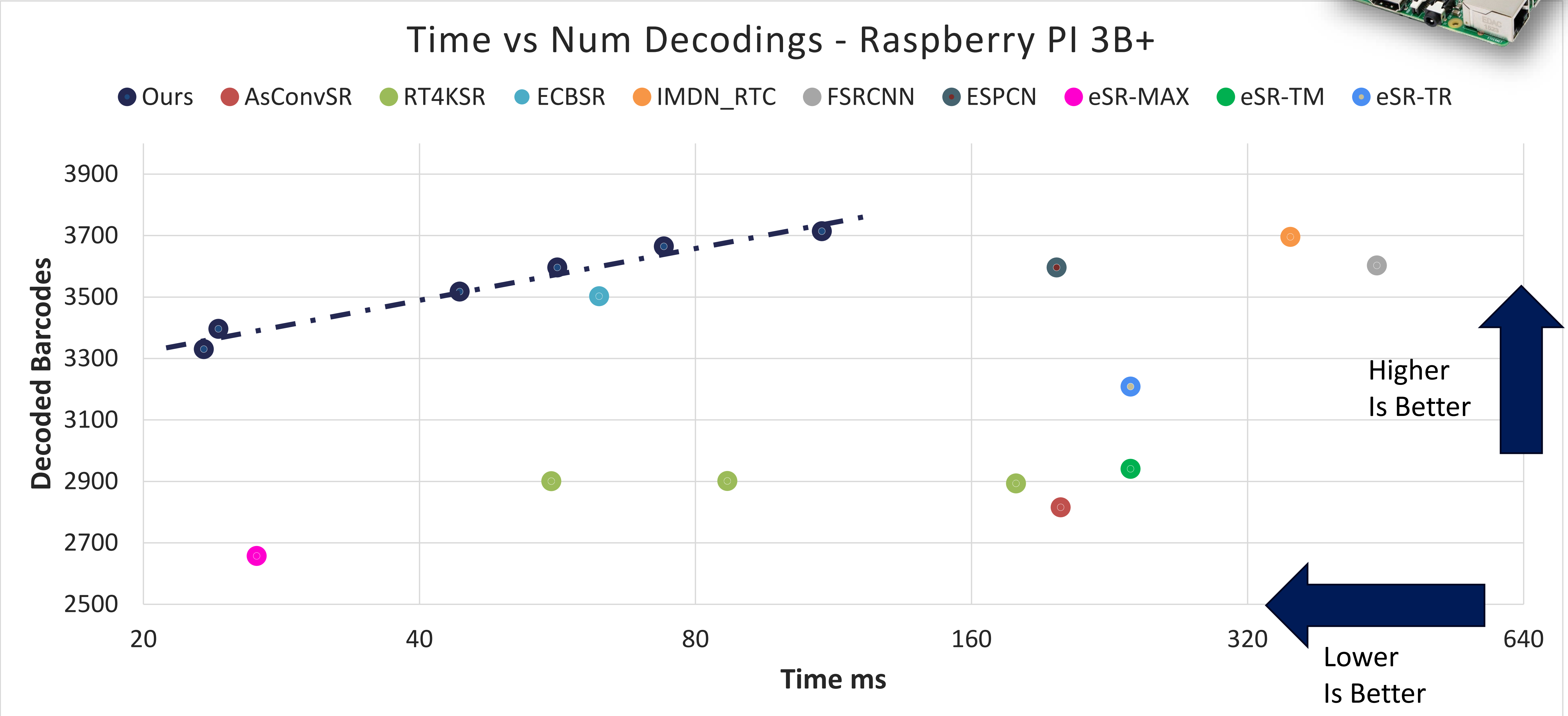
$\boldsymbol{M_3}$: 4 layers, 14,274 parameters

All modules have an input size of **7x7**

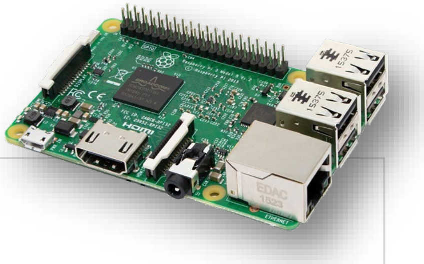


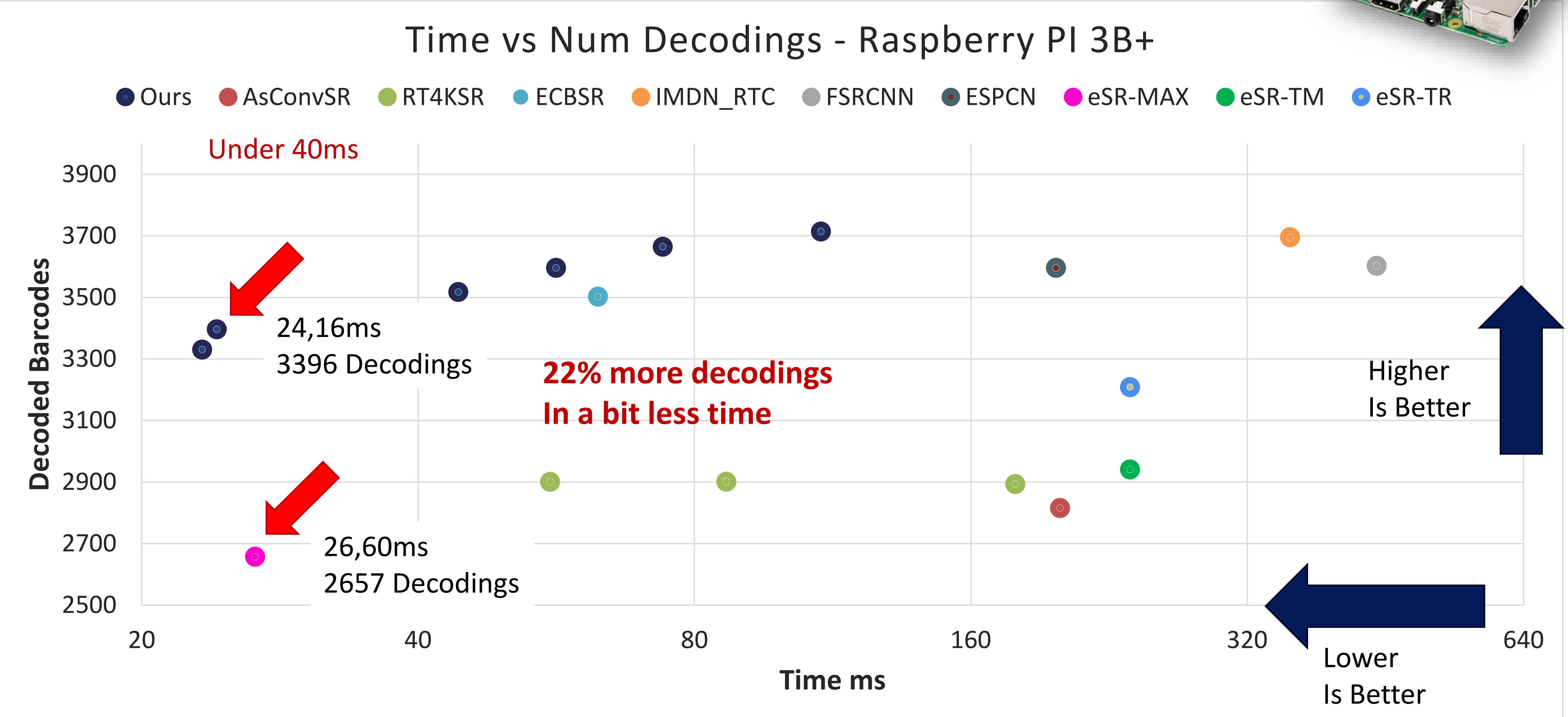
Test Results - BarBeR Dataset



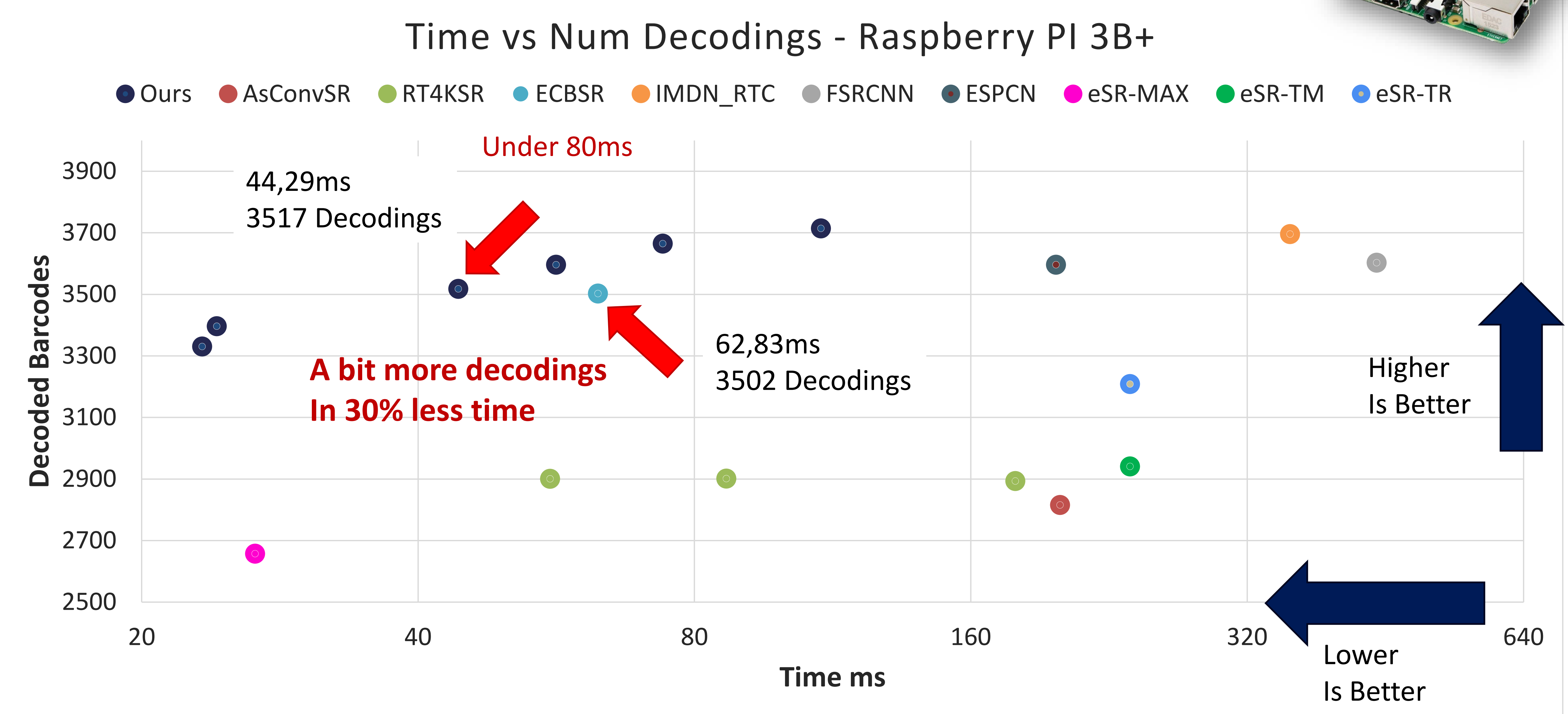


Test Results - BarBeR Dataset

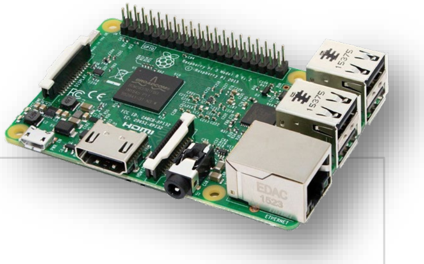


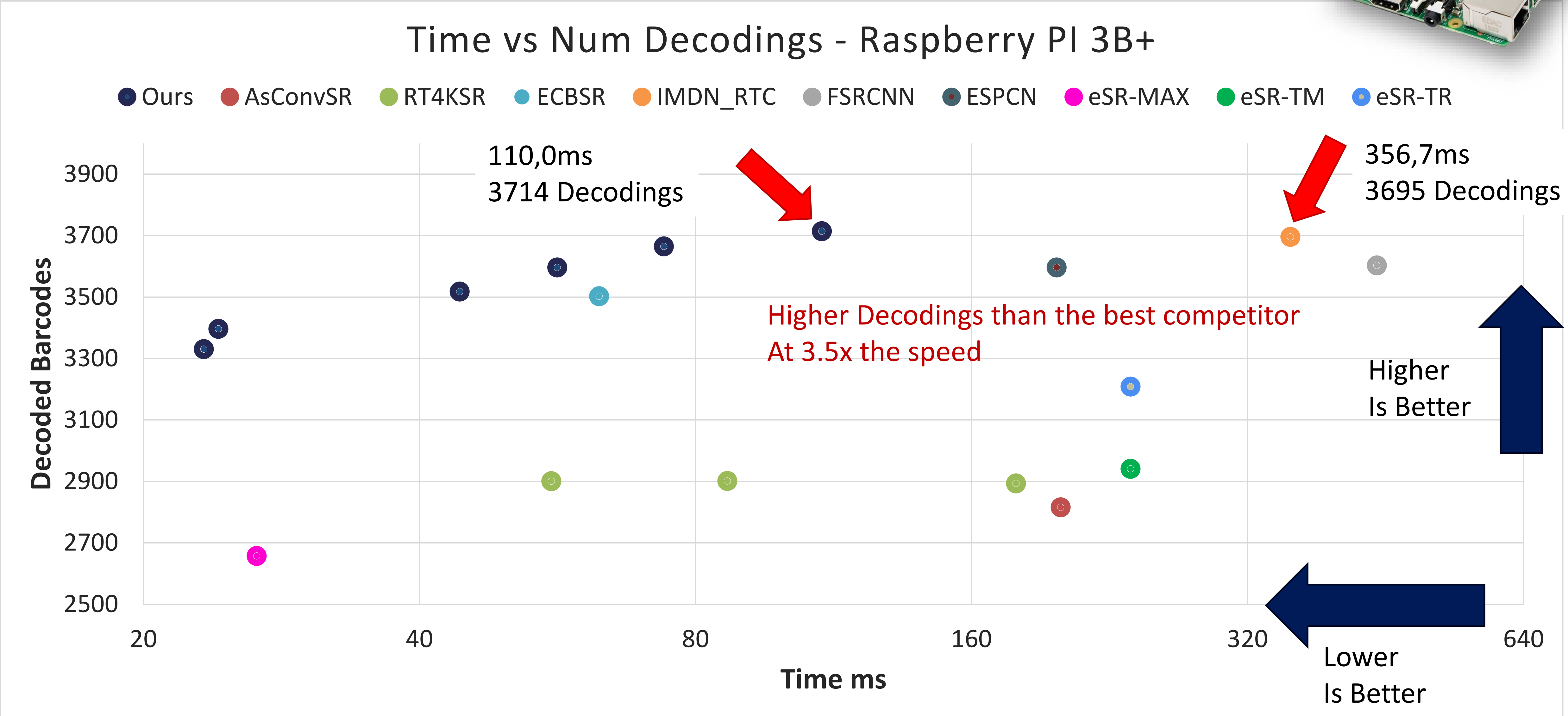


Test Results - BarBeR Dataset

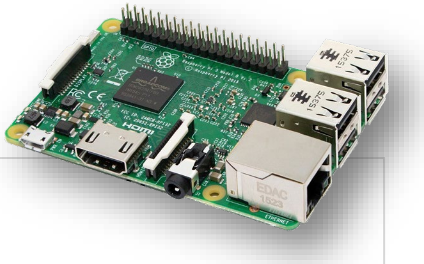


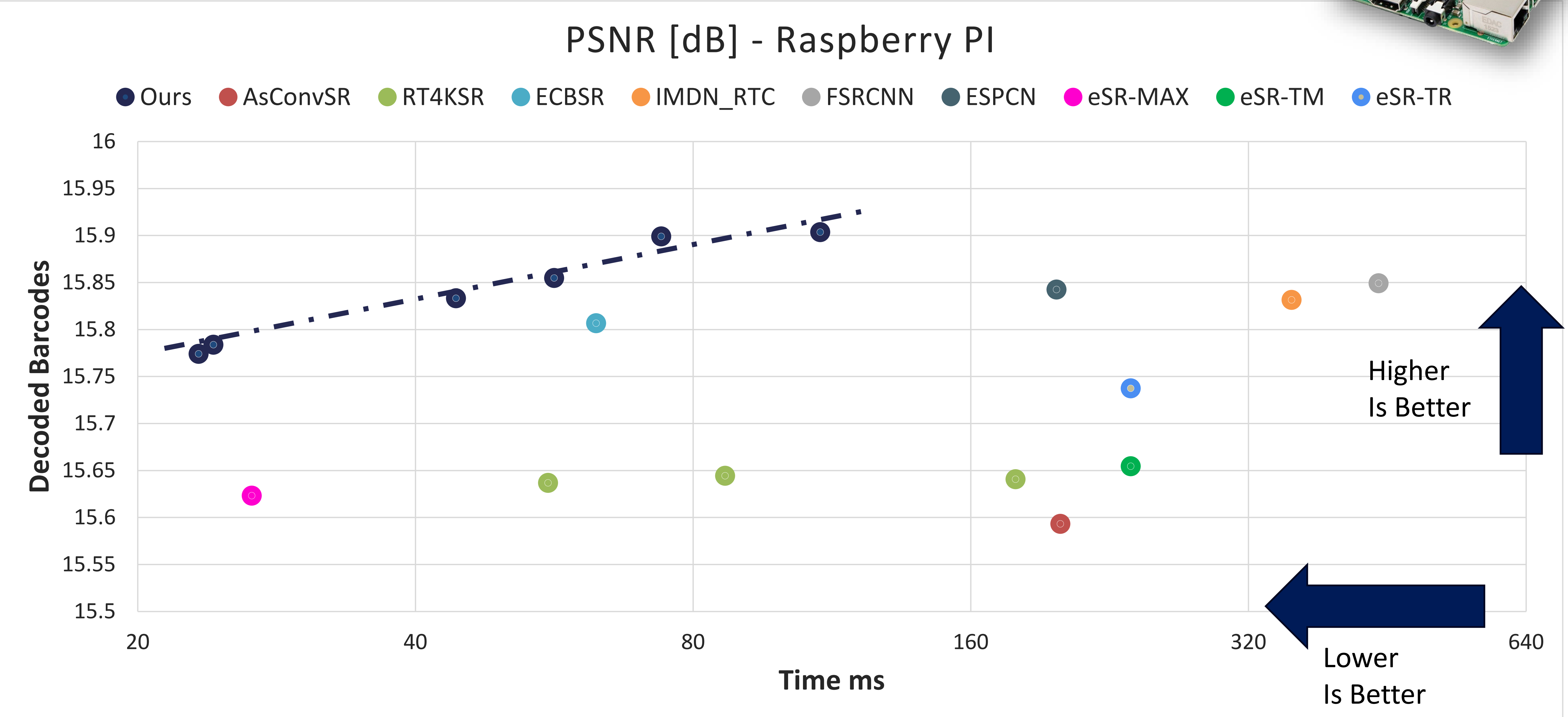
Test Results - BarBeR Dataset



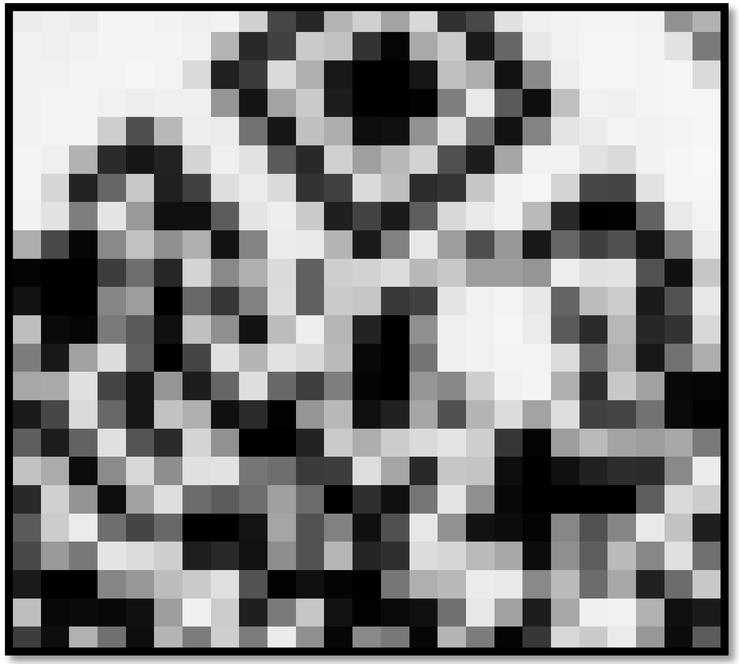


Test Results - BarBeR Dataset



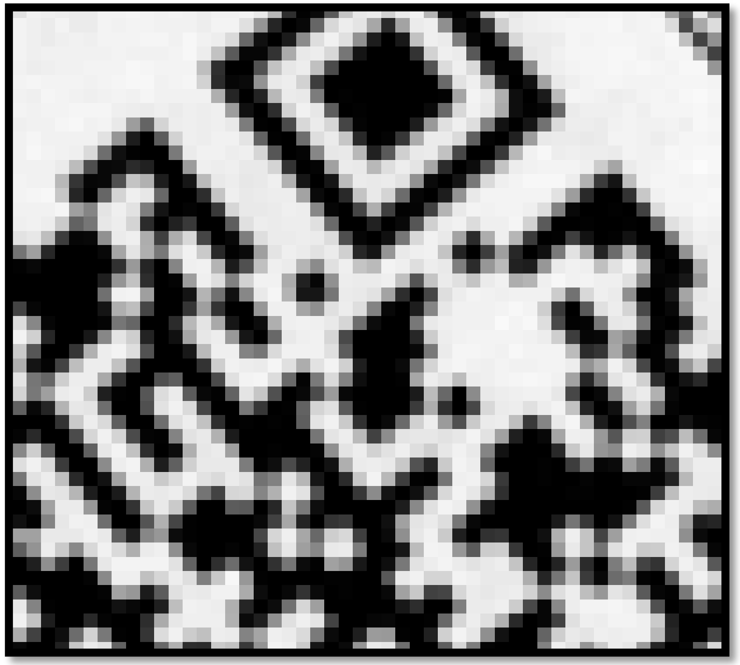


Visual Results – BarBeR



Low Definition

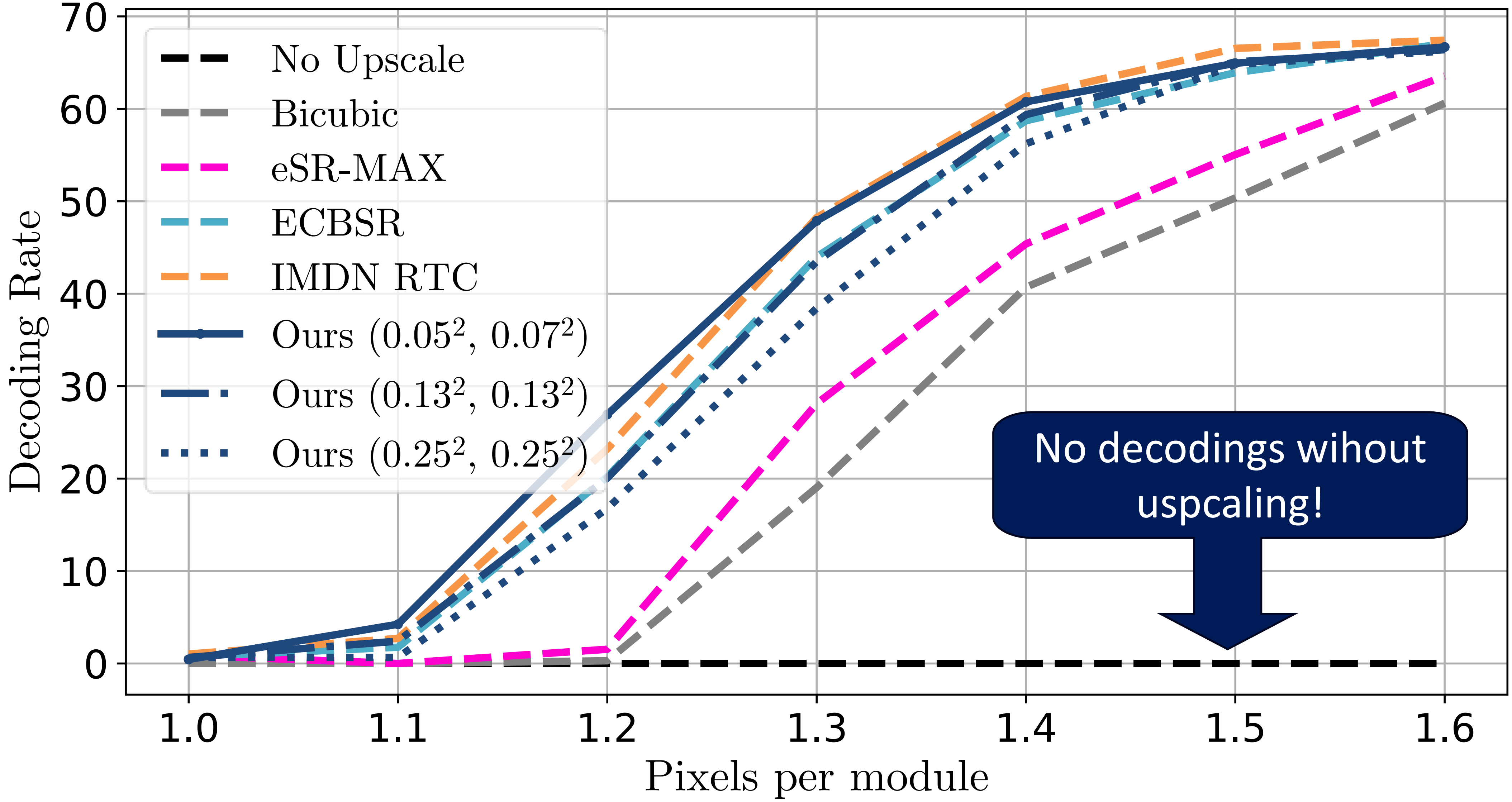
Ours – High Setting



HD Target

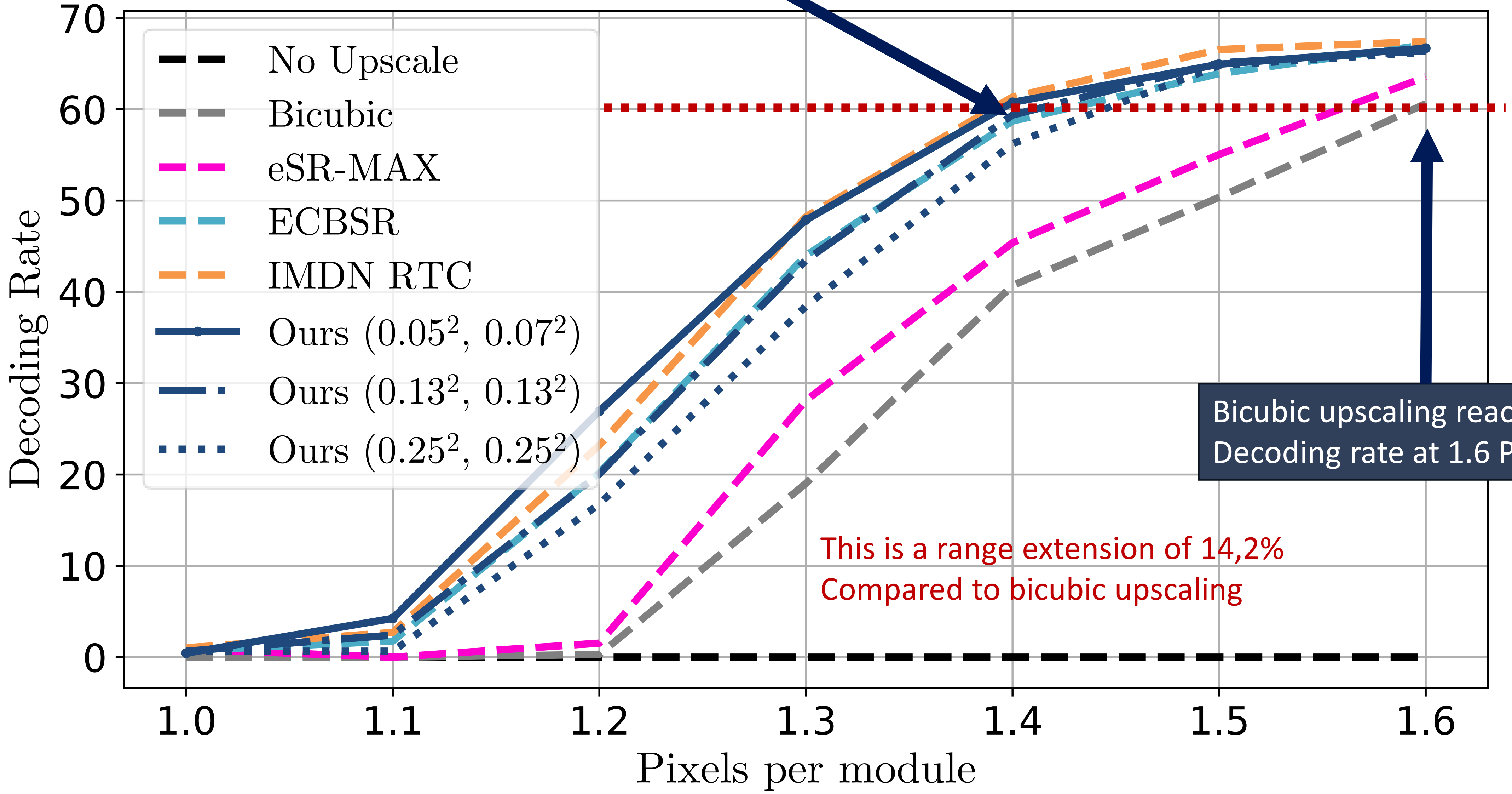
- **Example of a crop of a 1.3 PPM QR Code**
- **MosaicSR allows for correct classification between white and black modules in challenging images like this one**

Decoding Rate VS PPM



Decoding Rate VS PPM

Mosaic-SR reaches 60%
Decoding rate at 1.4 PPM



Bicubic upscaling reaches 60%
Decoding rate at 1.6 PPM

This is a range extension of 14,2%
Compared to bicubic upscaling

THANK YOU

LikedIn Contact

Mosaic-SR GitHub

Mail: enrico.vezzali3@gmail.com