CSAIL

Neural Inverse Knitting: From Images to Manufacturing Instructions

Alexandre Kaspar*, Tae-Hyun Oh*, Liane Makatura,
Petr Kellnhofer and Wojciech Matusik

Industrial Knitting

Example: Sweater

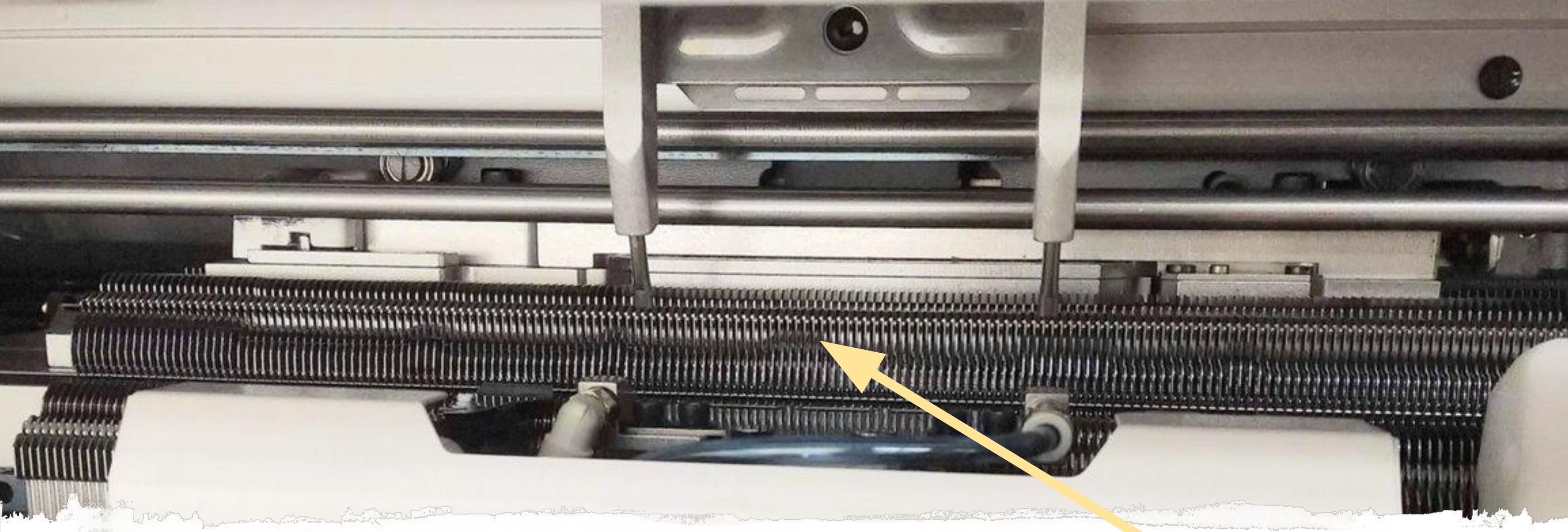
[Ministry of Supply]

Example: Scarf

[Kniterate]

Industrial Knitting

- Whole garments from scratch



Industrial Knitting

- Control of individual needles
- Whole garments from scratch

Knitted Garment & Patterns

Many garments are knitted:

- Beanies, scarves
- Gloves, socks and underwear
- Sweaters, sweatpants

Current machines can create those garments **seamlessly** (no sewing needed).












Knitted Garment & Patterns

Those garments have **various types** of surface **patterns** (knitting patterns).

These can be fully controlled by industrial knitting machine.

= User customization!

Machine Knitting Programming

Low-level machine code
requires skilled experts

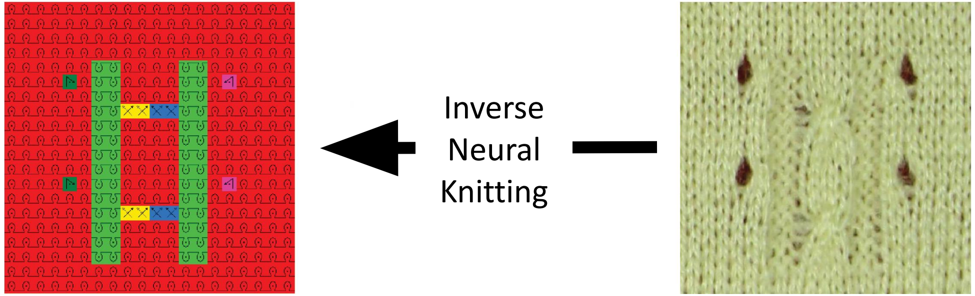
= Knitting masters

Scenario

1. User takes picture of knitting pattern

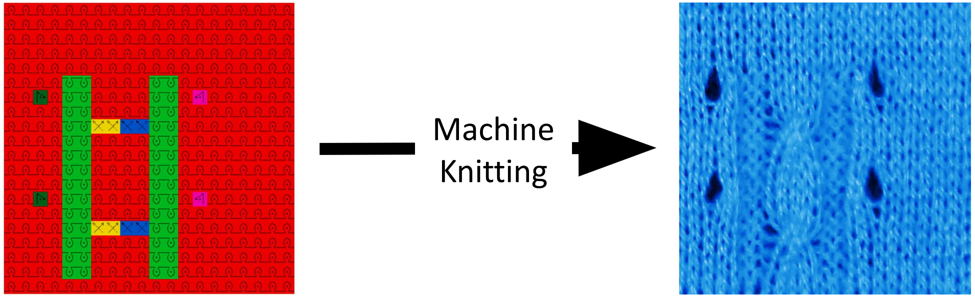
Scenario

1. User takes picture of knitting pattern
2. System creates knitting instructions



Scenario

1. User takes picture of knitting pattern
2. System creates knitting instructions
3. User reuses pattern for new garment



Machine Knitting

Brief background

Machine Knitting Terminology

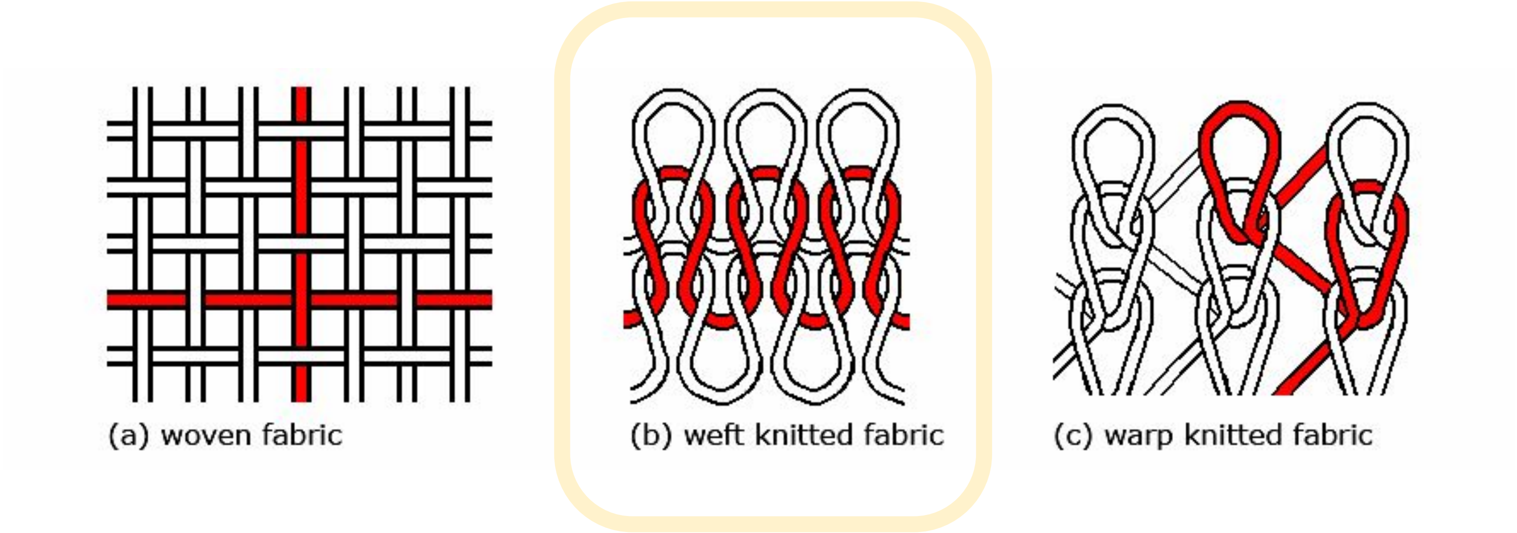


Illustration from [Underwood09]

Machine Knitting Terminology

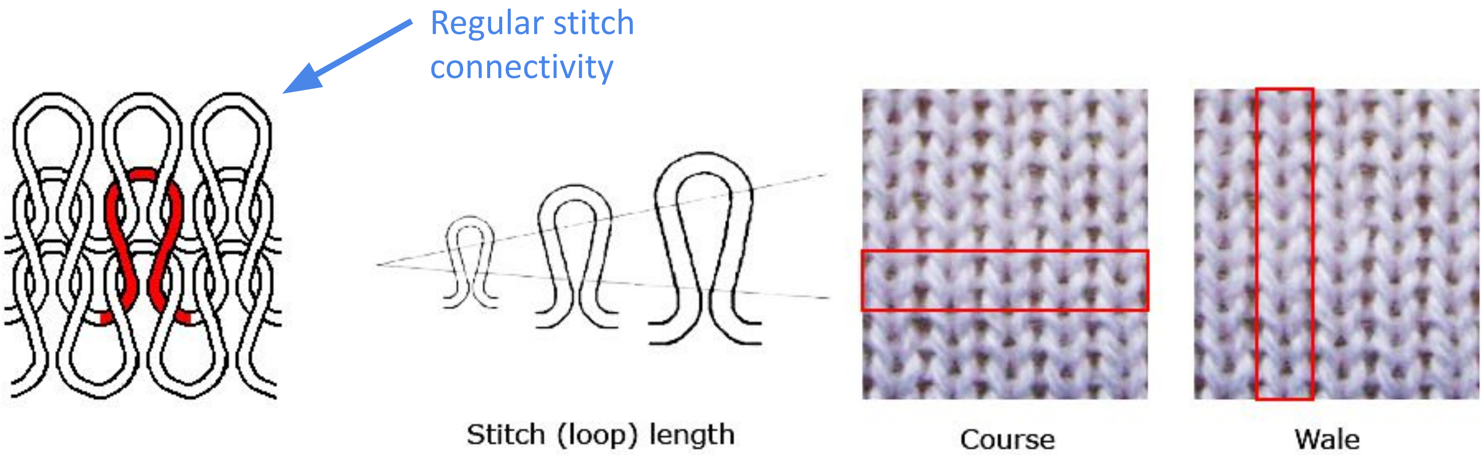


Illustration from [Underwood09]

Machine Knitting Terminology

V-bed machine & knitting bed

Machine Knitting Terminology

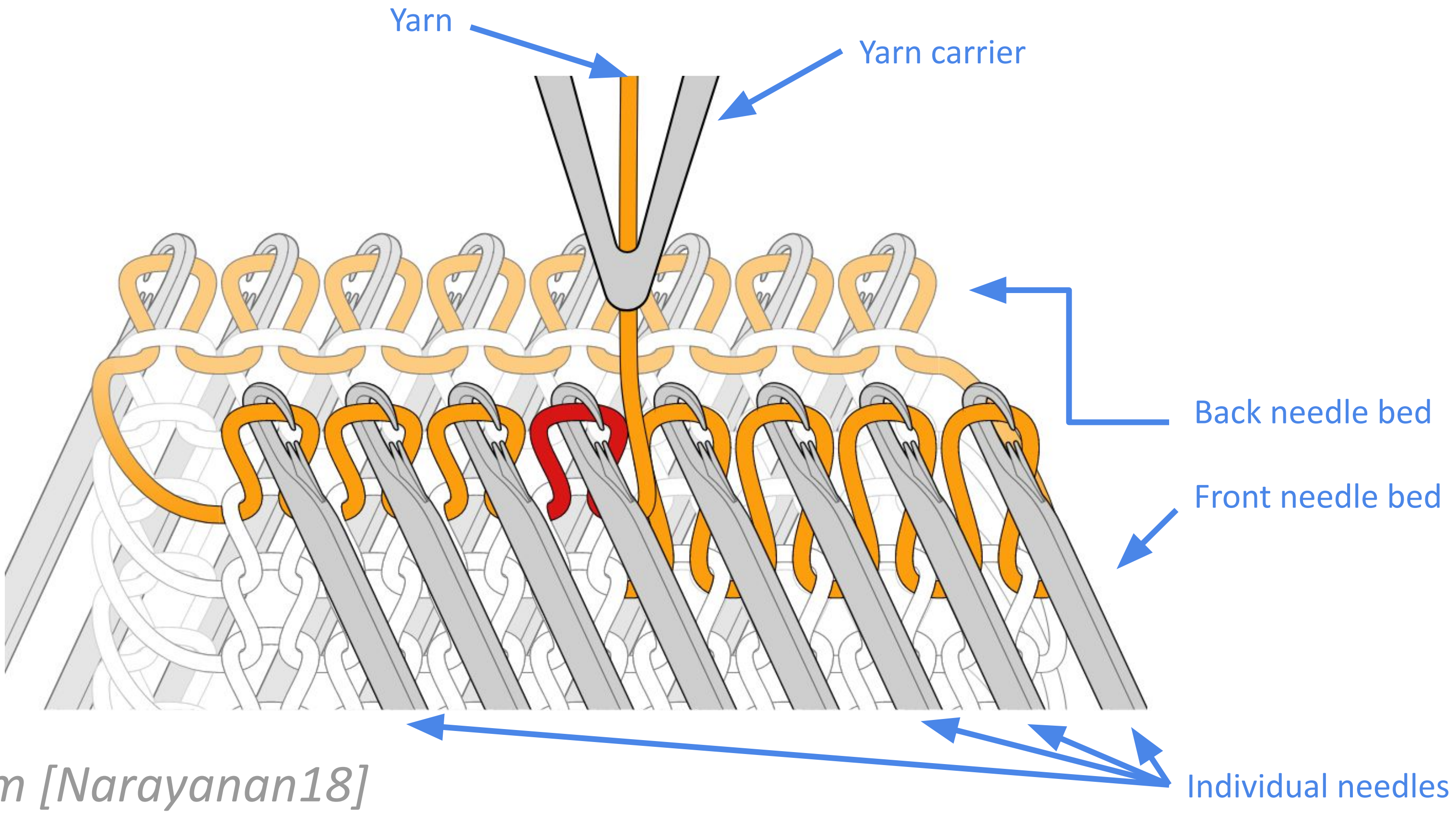


Illustration from [Narayanan18]

Machine Knitting Operations

"Tuck" operation

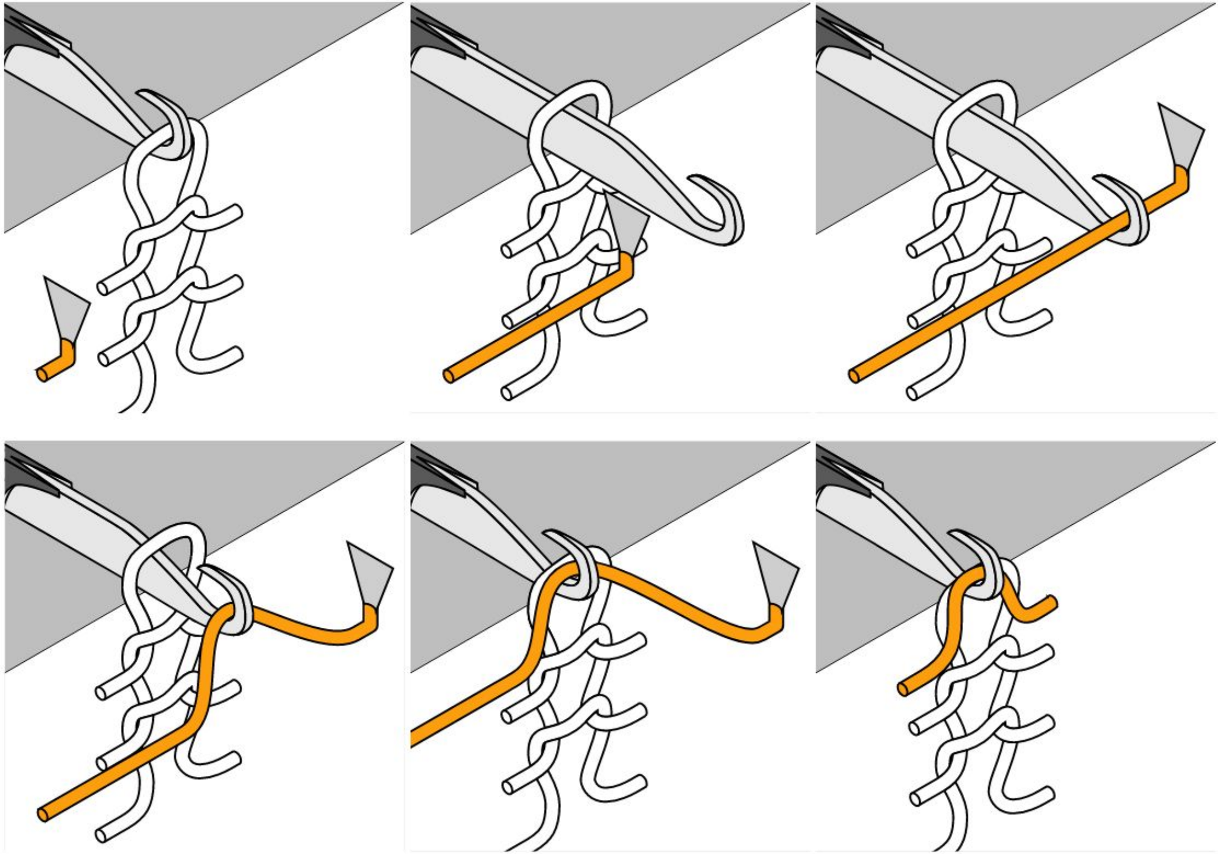


Illustration from [Narayanan18]

Machine Knitting Operations

“Knit” operation

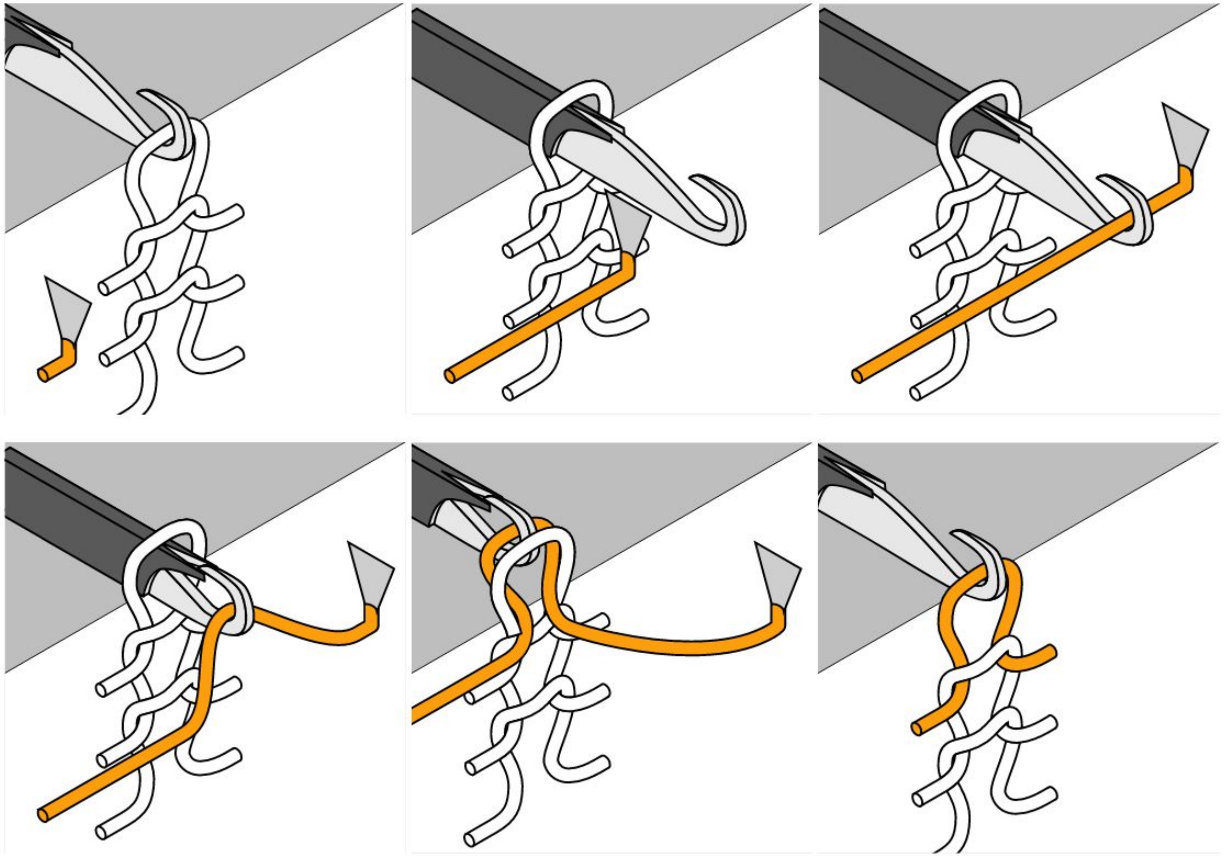


Illustration from [Narayanan18]

Machine Knitting Operations

“Transfer” operation

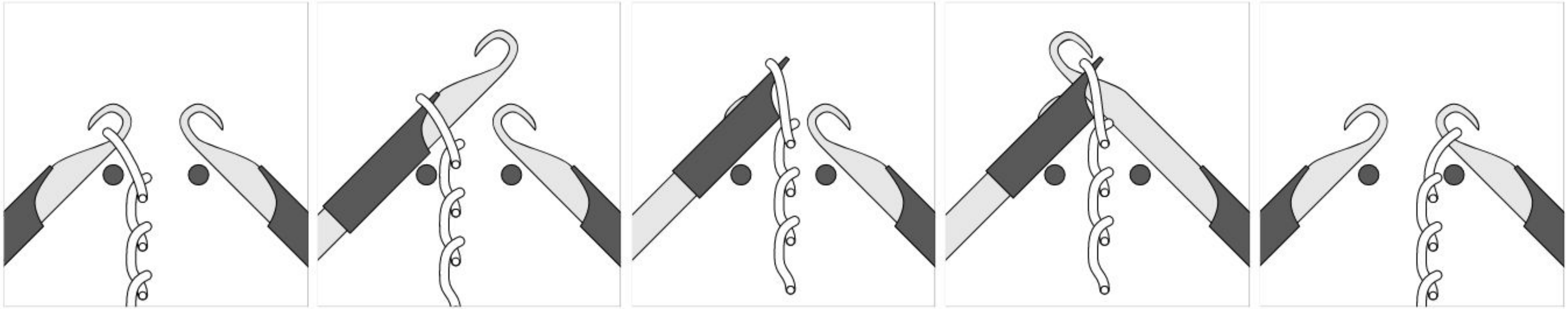
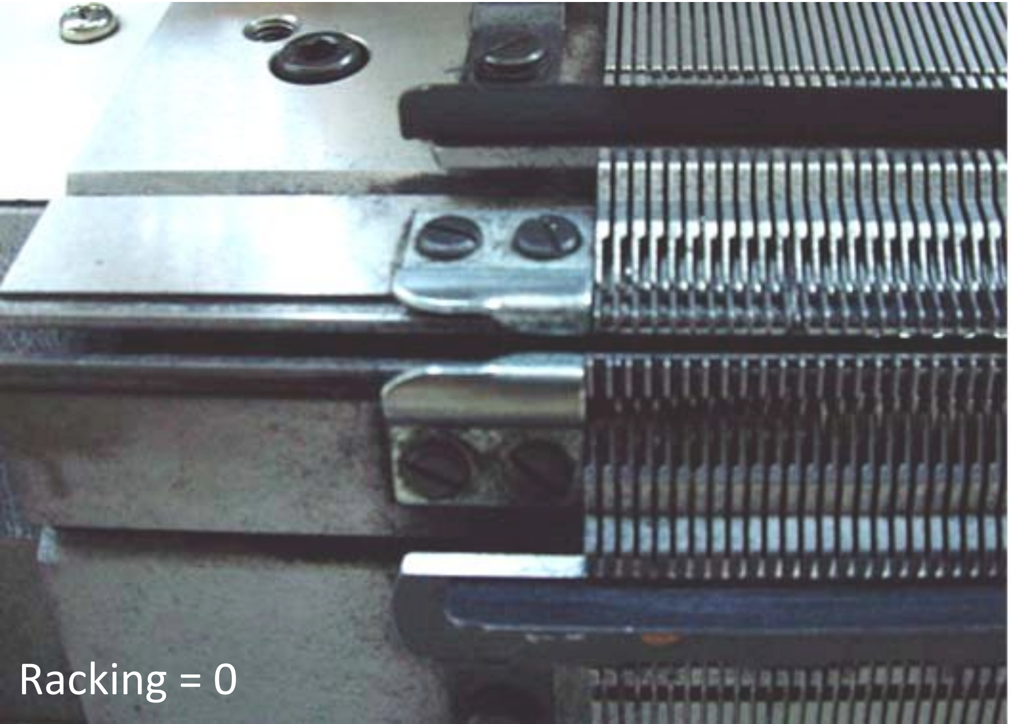
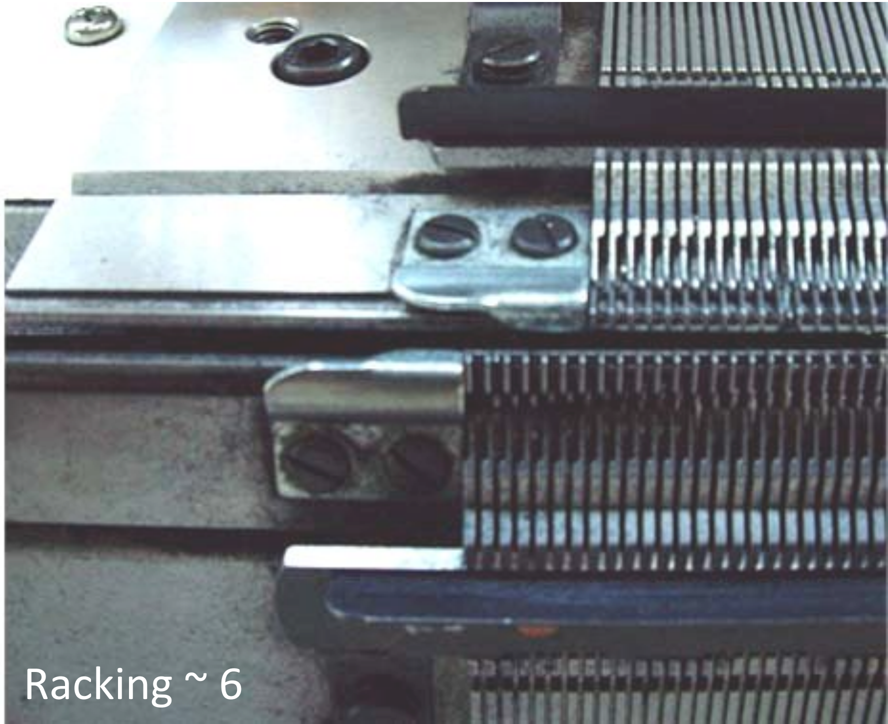


Illustration from [Narayanan18]

Machine Knitting Operations

Racking = offsetting between the two beds



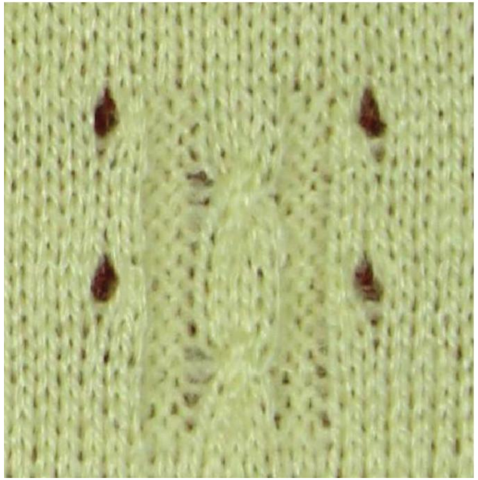


The Data & its Acquisition

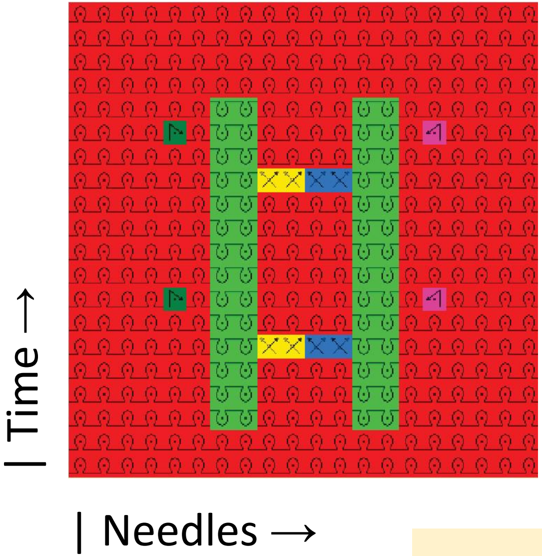
For 2D machine knitting programs

2D Knitting Pattern Programs

Image of 20x20 pattern



20x20 pattern program



"Pixels" are per-needle instructions over time

Knitting Pattern DSL

Domain Specific Language (DSL) for regular knitting patterns

K P T M

Basic operations

XR+ XR- XL+ XL-

Cross operations

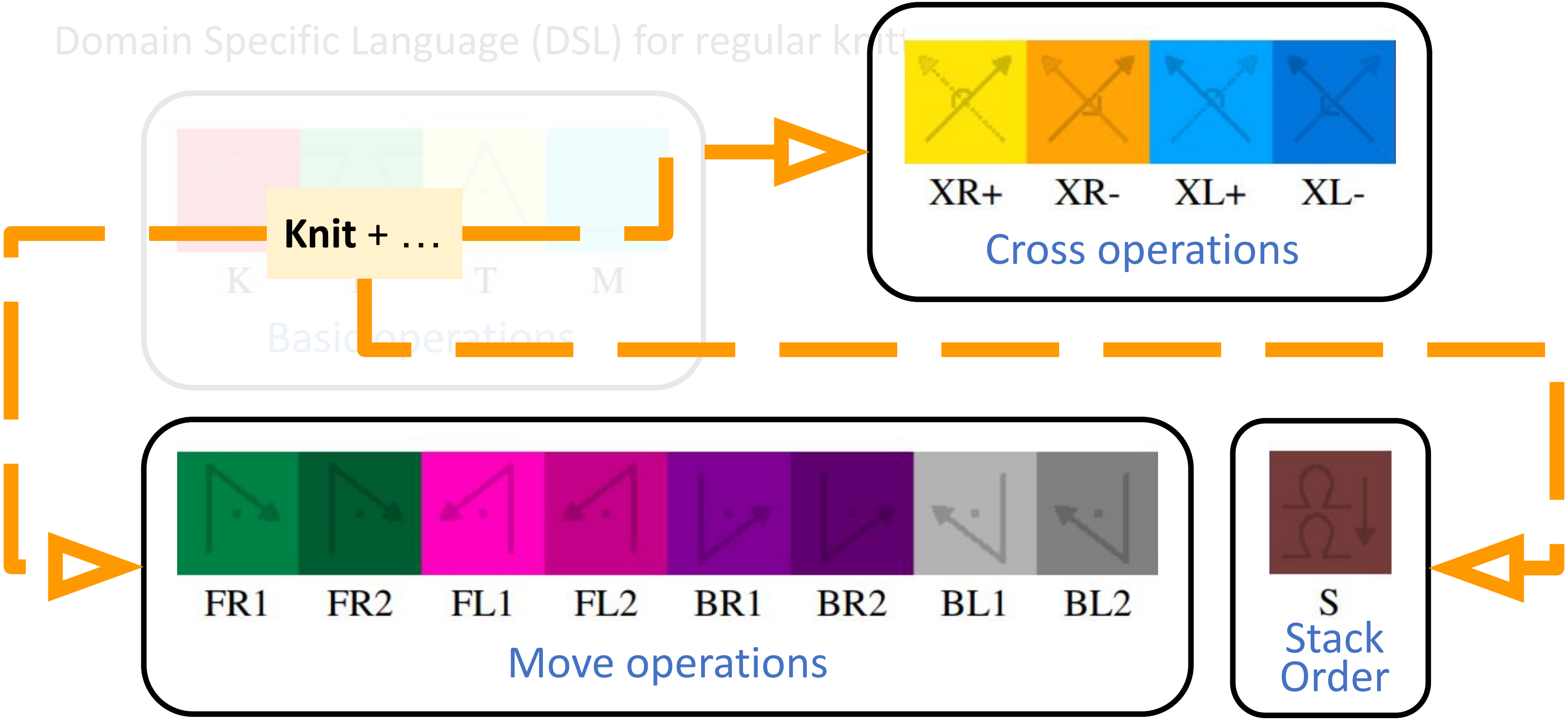
FR1 FR2 FL1 FL2 BR1 BR2 BL1 BL2

Move operations

S

Stack Order

Knitting Pattern DSL



DSL: from regular grids to sequences

Full rows of operations are executed at once with the following sequence:

1. Move “current stitches” to the operation side (front | back)
2. Apply “needle operation” (knit | tuck | miss)
3. Transfer moving stitches to back bed (cross | move | stack)
4. Apply *sequence* of moves depending on the operations (cross | move)
5. Bring back all stitches to front bed (purl | cross | move | stack)

DSL: from regular grids to sequences

Full rows of operations are executed at once with the following sequence:

1. Move "current stitches" to the operation side (front | back)
2. Apply "needle operation" (knit | tuck | miss)
3. Transfer moving stitches to back bed (cross | move | stack)
4. Apply *sequence* of operation-related moves (cross | move)
5. Bring back all stitches to front bed (purl | cross | move | stack)

Encoded by operation type:

- **Move** = relative order not important
- **Cross** = relative order defined by group and "order" (upper | lower)

Dataset: Initial Attempt

Individual 20x20 patterns

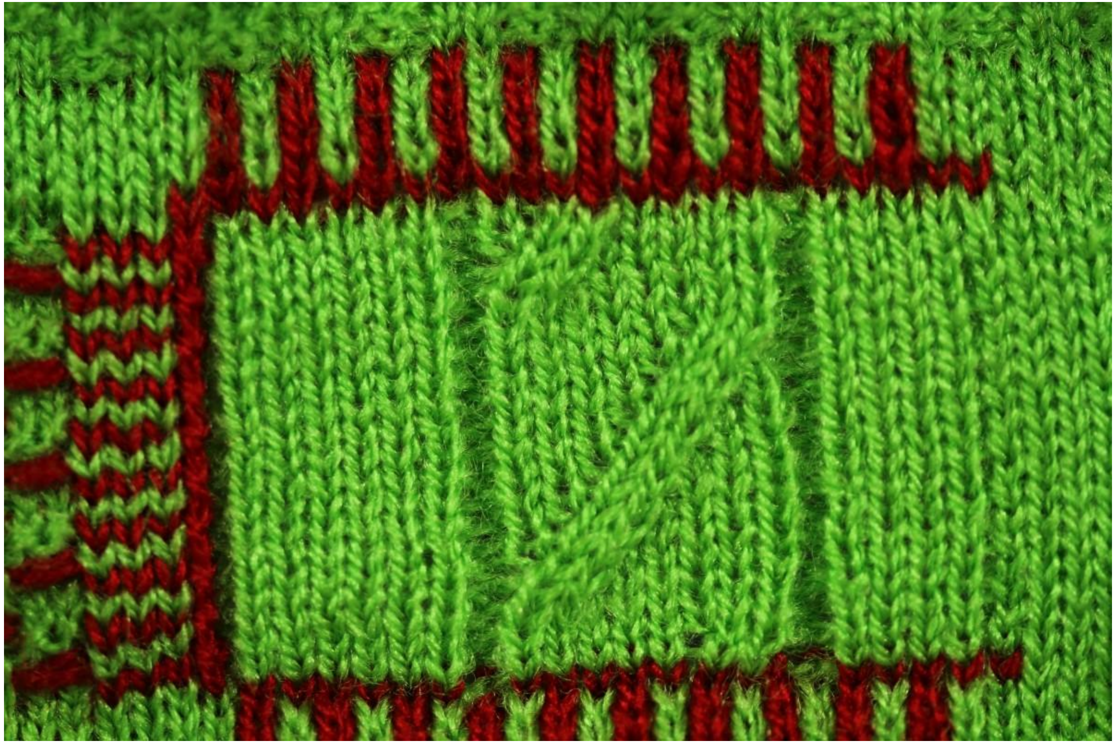
High-quality registration

- From color frame
- Still not per-stitch…

Total: ~200 patterns

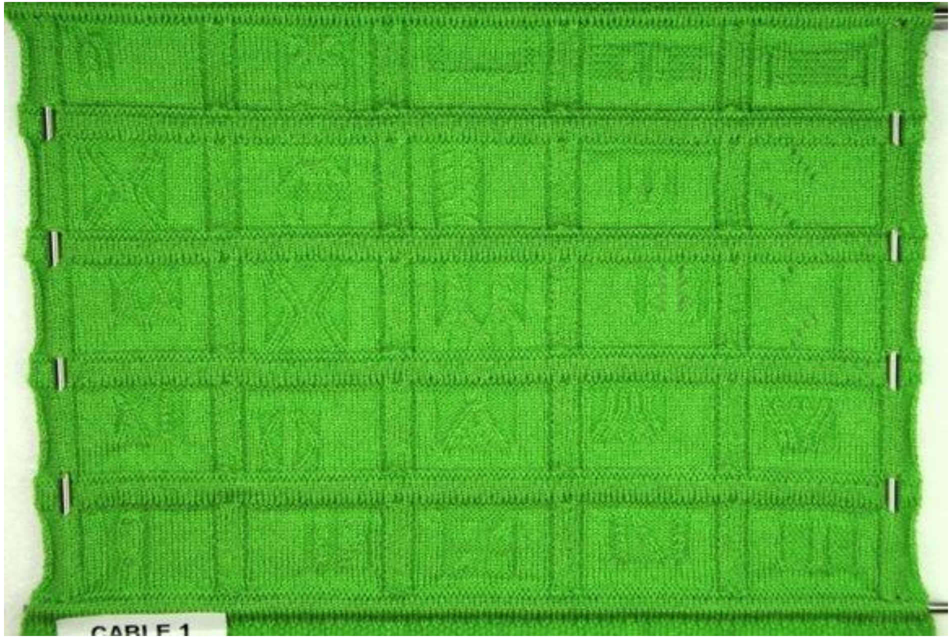
Time: ~1 month (intern)

= not enough data!



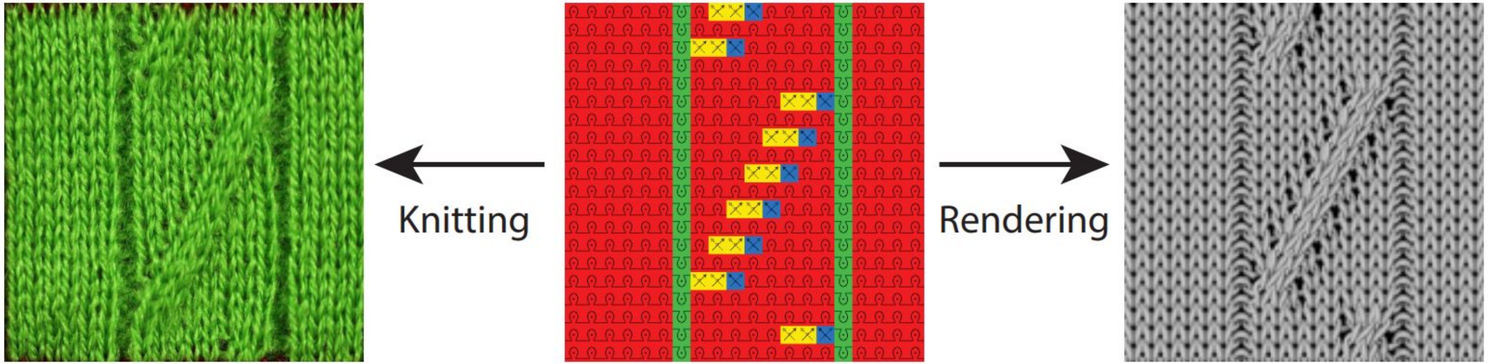
Dataset: Better Attempt

Capture setup with steel rods to normalize tension



Dataset Content

- Paired instructions with real (2,088) and synthetic (14,440) images.
- Synthetic data from automatic screen capture of KnitPaint (Shima's software)



Machine Learning Details

Using two different types of supervision data

Learning Problem

Mapping **images** to discrete **instruction maps**

= CE loss minimization

Using two domains of data (one real, one synthetic)

= How to best combine both

Generalization Bound with Two Domains

With probability at least $1 - \delta$

$$\frac{1}{2}\underbrace{\left|\mathcal{L}_T(\hat{h}, y) - \mathcal{L}_T(h_T^*, y)\right|}_{\text{Generalization gap}} \leq \alpha\left(\mathrm{disc}_{\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T) + \lambda\right) + \epsilon$$

Ideal min.

Generalization Bound with Two Domains

With probability at least $1-\delta$

$$\underbrace{\frac{1}{2}\left|\mathcal{L}_T(\hat{h},y)-\mathcal{L}_T(h_T^*,y)\right|}_{\text{Generalization gap}} \leq \alpha\left(\mathrm{disc}_{\mathcal{H}}(\mathcal{D}_S,\mathcal{D}_T)+\lambda\right)+\epsilon$$

Ideal min.

Empirical min. $\arg\min_h \alpha\mathcal{L}_{\hat{S}}(h,y)+(1-\alpha)\mathcal{L}_{\hat{T}}(h,y)$

Generalization Bound with Two Domains

With probability at least $1 - \delta$

$$\frac{1}{2}|\mathcal{L}_T(\hat{h}, y) - \mathcal{L}_T(h_T^*, y)| \leq \alpha\left(\mathrm{disc}_{\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T) + \lambda\right) + \epsilon$$

Generalization Bound with Two Domains

With probability at least $1-\delta$

$$\frac{1}{2}|\mathcal{L}_T(\hat{h},y) - \mathcal{L}_T(h_T^*,y)|$$

$$\leq \alpha\left(\mathrm{disc}_{\mathcal{H}}(\mathcal{D}_S,\mathcal{D}_T)+\lambda\right) + \epsilon$$

$$\boxed{\epsilon(m,\alpha,\beta,\delta) = \sqrt{\frac{1}{2m}\left(\frac{\alpha^2}{\beta}+\frac{(1-\alpha)^2}{1-\beta}\right)\log(\frac{2}{\delta})},}$$

Hyper-parameter dependent term

Generalization Bound with Two Domains

With probability at least $1-\delta$

$$\frac{1}{2}\left|\mathcal{L}_T(\hat{h},y)-\mathcal{L}_T(h_T^*,y)\right|$$

$$\leq \alpha\left(\mathrm{disc}_{\mathcal{H}}(\mathcal{D}_S,\mathcal{D}_T)+\lambda\right)+\epsilon$$

$$\boxed{\lambda=\min_{h\in\mathcal{H}}\mathcal{L}_S(h,y)+\mathcal{L}_T(h,y).}$$

Ideal error of the combined losses

Generalization Bound with Two Domains

With probability at least $1 - \delta$

$$\frac{1}{2}\left|\mathcal{L}_T(\hat{h}, y) - \mathcal{L}_T(h_T^*, y)\right|$$

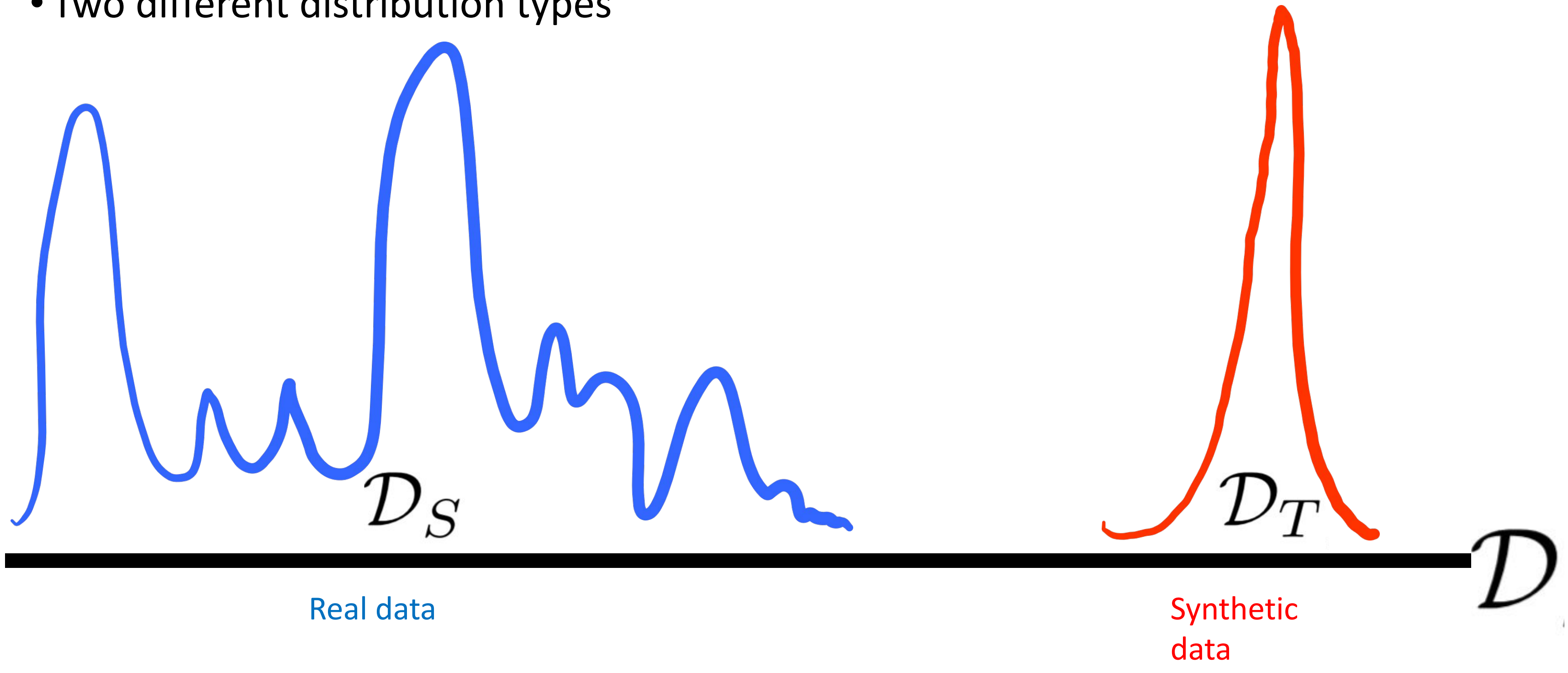
$$\leq \alpha\left(\boxed{\mathrm{disc}_{\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T)} + \lambda\right) + \epsilon$$

Discrepancy between distributions

$$\boxed{\mathrm{disc}_{\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T)} = \max_{h, h' \in \mathcal{H}} \left|\mathcal{L}_{\mathcal{D}_S}(h, h') - \mathcal{L}_{\mathcal{D}_T}(h, h')\right|$$

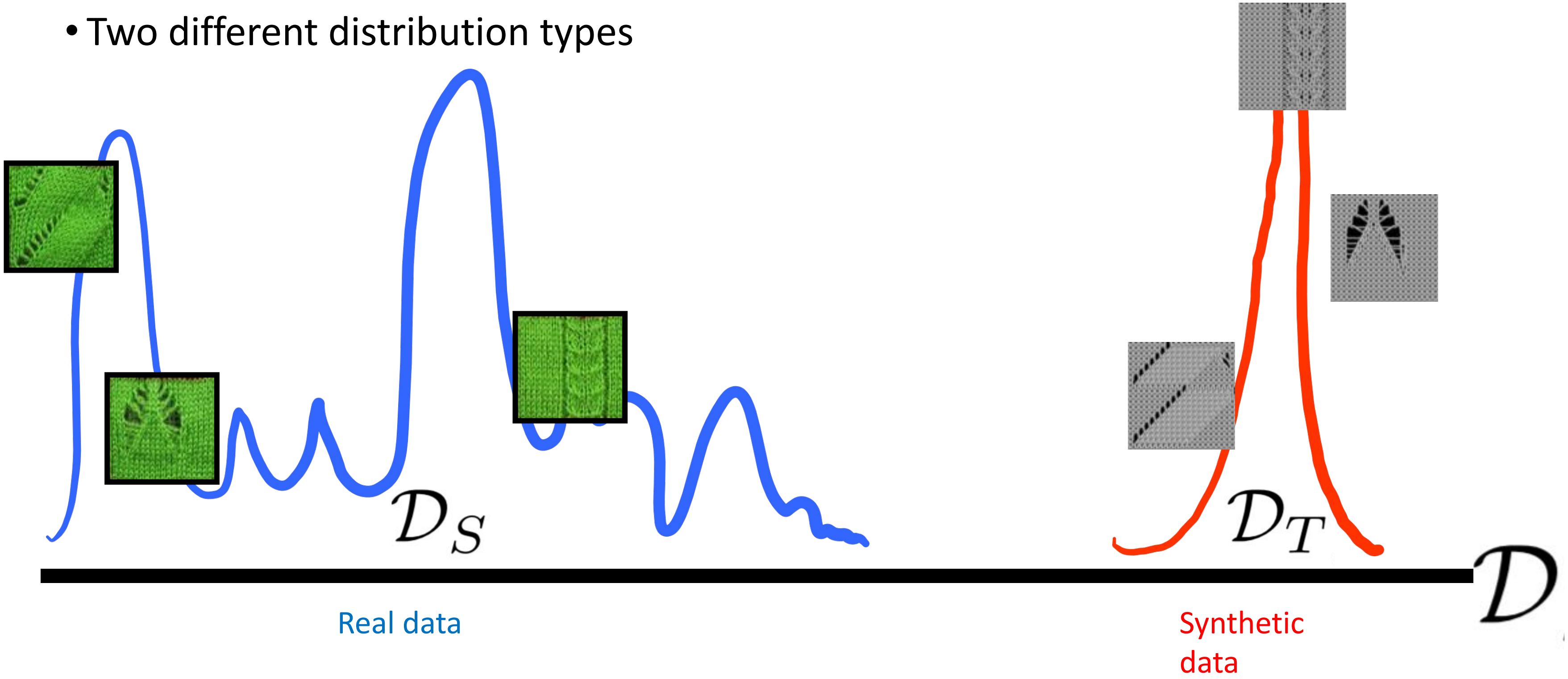
Data distributions

- Two different distribution types



Data distributions

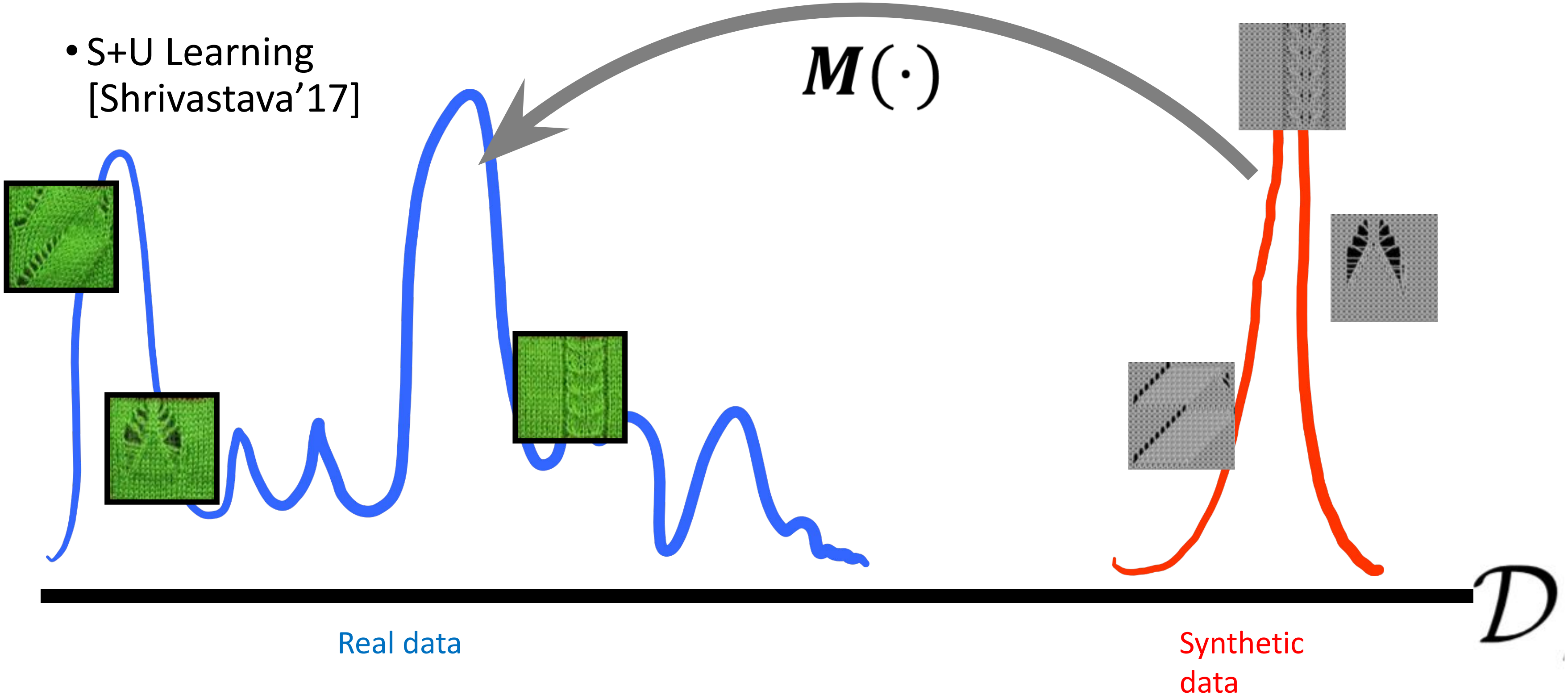
- Two different distribution types



From synthetic to real

$$\min_{M} \mathrm{disc}(\mathcal{D}_S, \underset{S \Leftarrow T}{M(\mathcal{D}_T)})$$

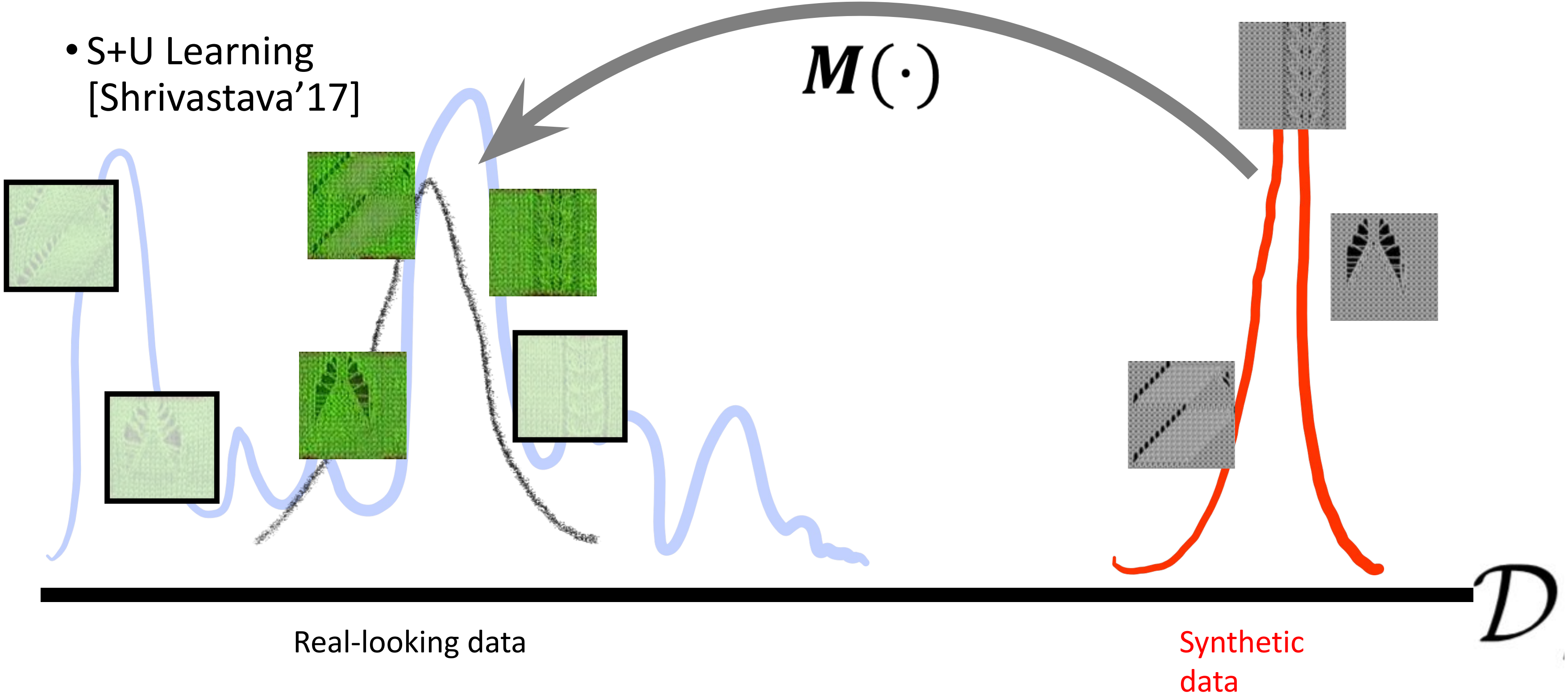
- S+U Learning [Shrivastava'17]



From synthetic to real

$$\min_{M} \operatorname{disc}(\mathcal{D}_S, \underset{S \Leftarrow T}{M(\mathcal{D}_T)})$$

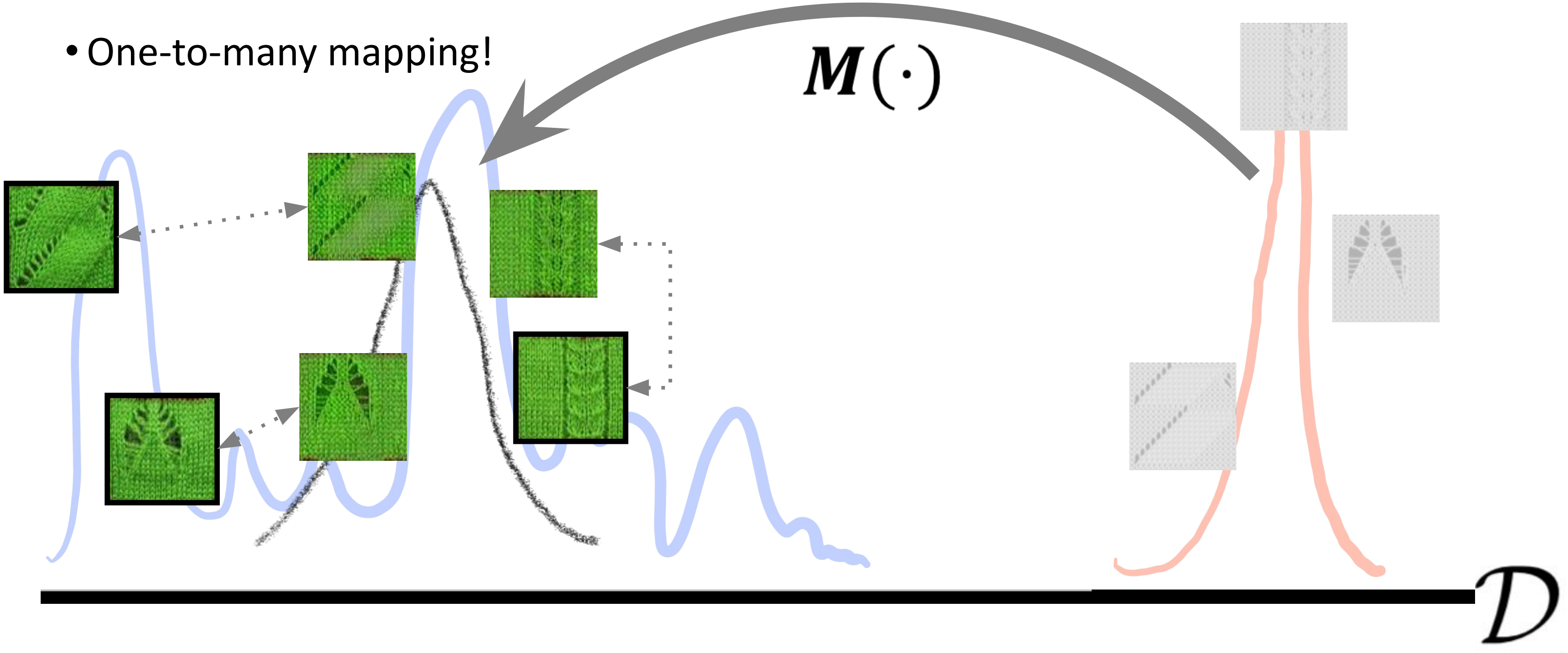
- S+U Learning [Shrivastava'17]



From synthetic to real

$$\min_{M} \operatorname{disc}(\mathcal{D}_S, \underbrace{M(\mathcal{D}_T)}_{S \Leftarrow T})$$

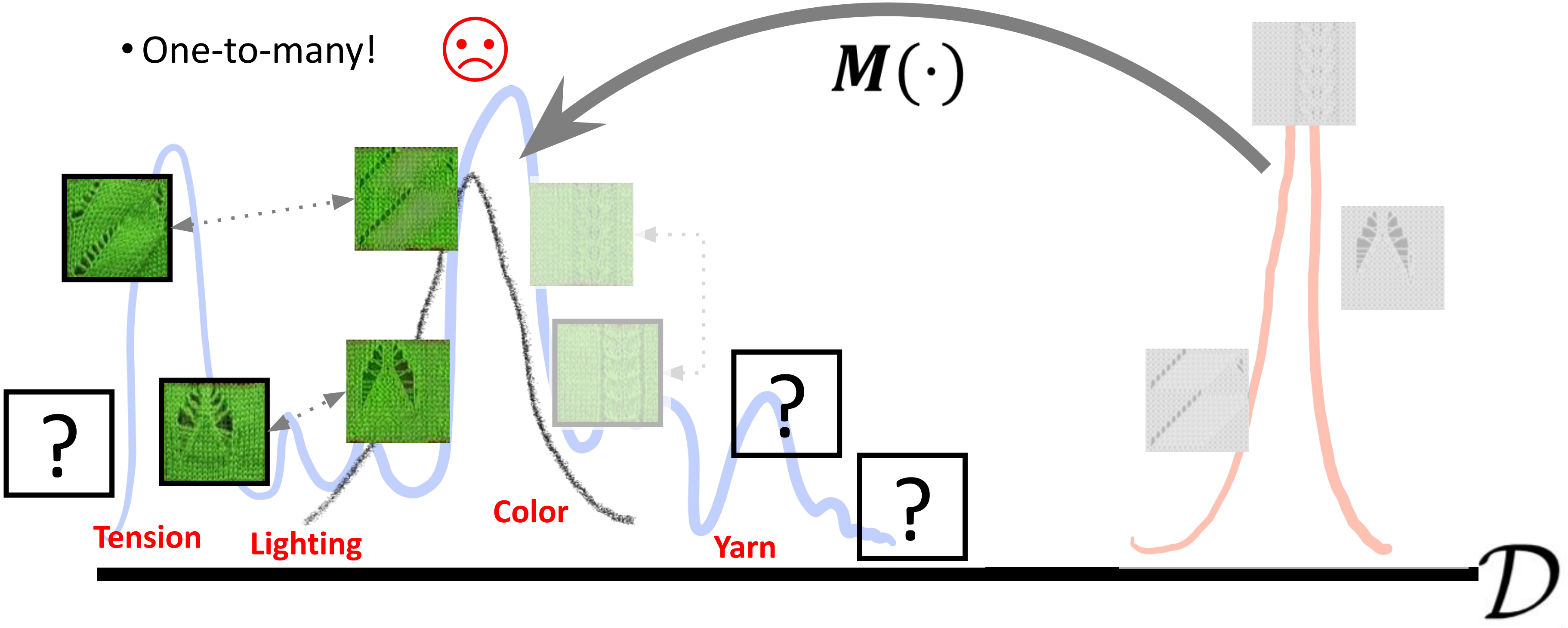
- One-to-many mapping!



From synthetic to real

$$\min_{M} \operatorname{disc}(\mathcal{D}_S, \underbrace{M(\mathcal{D}_T)}_{S \Leftarrow T})$$

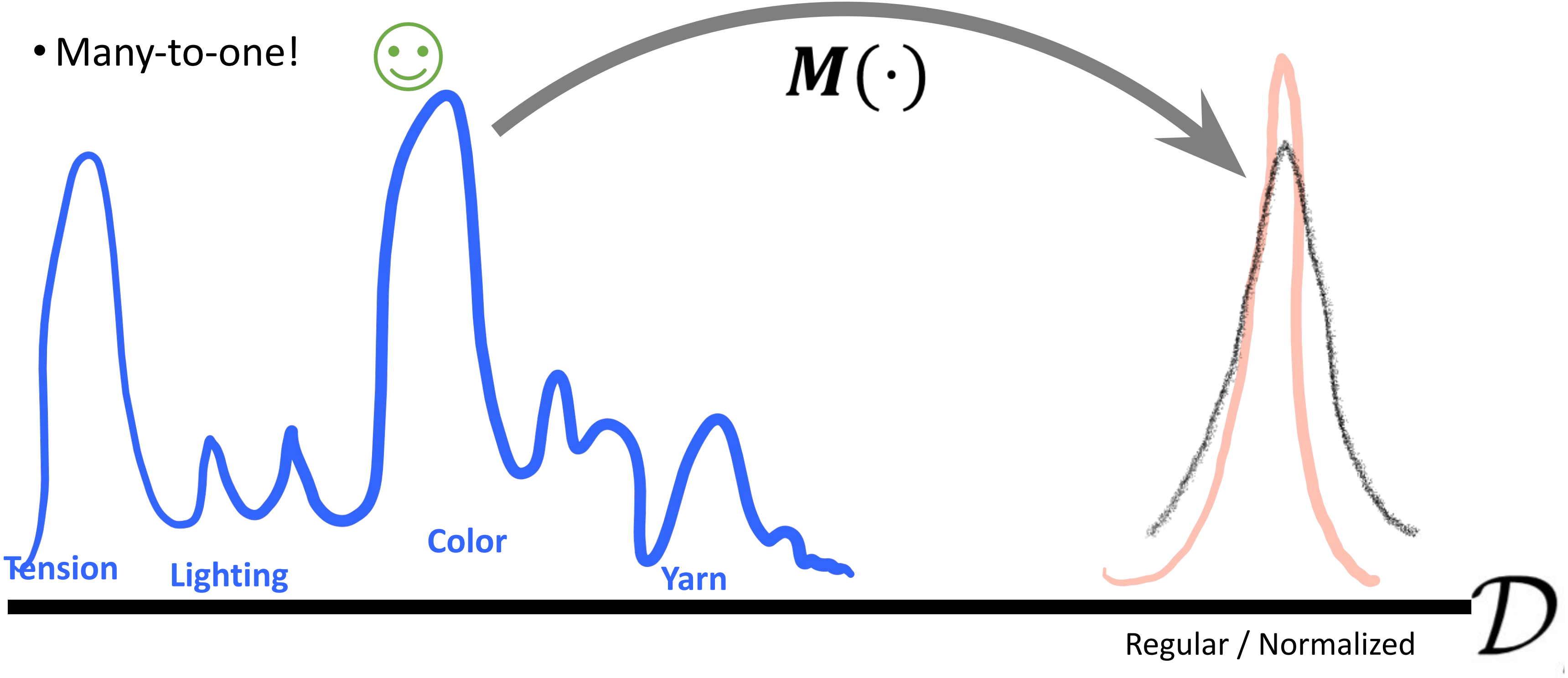
- One-to-many! ☹



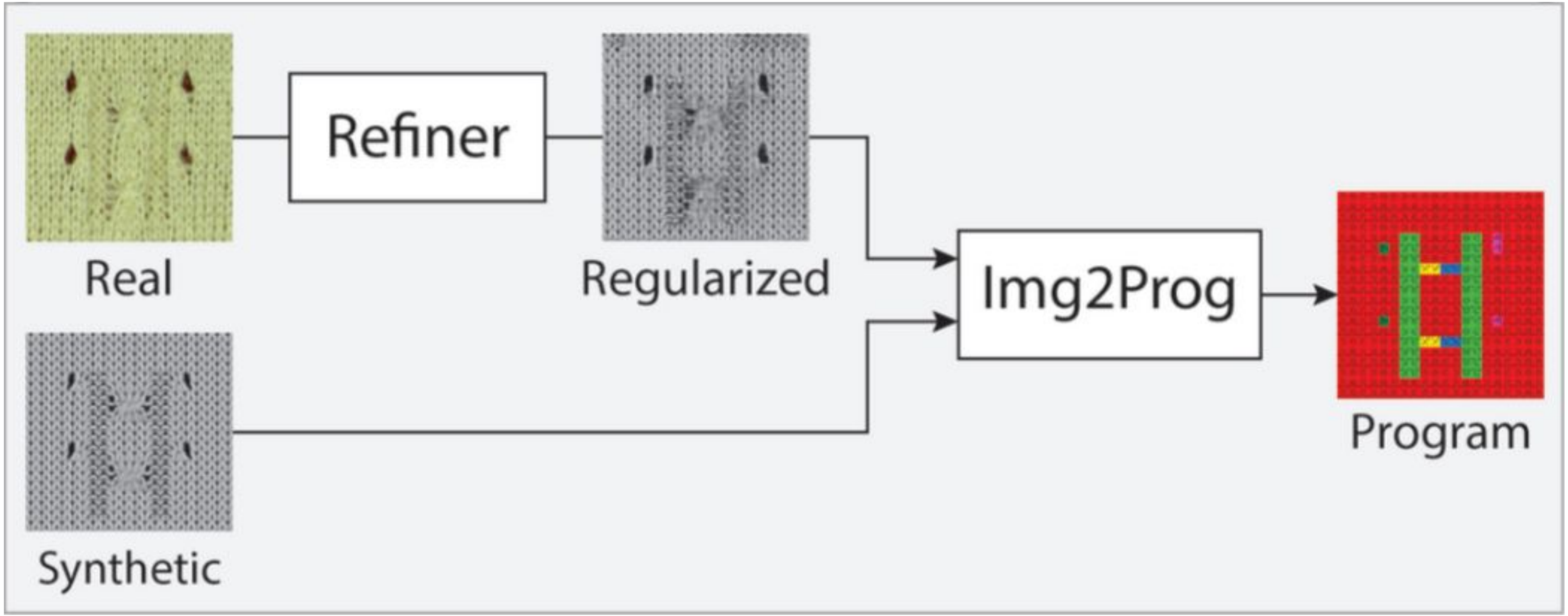
From real to synthetic

$$\min_{M} \operatorname{disc}_{S \Rightarrow T}(M(\mathcal{D}_S), \mathcal{D}_T)$$

- Many-to-one!

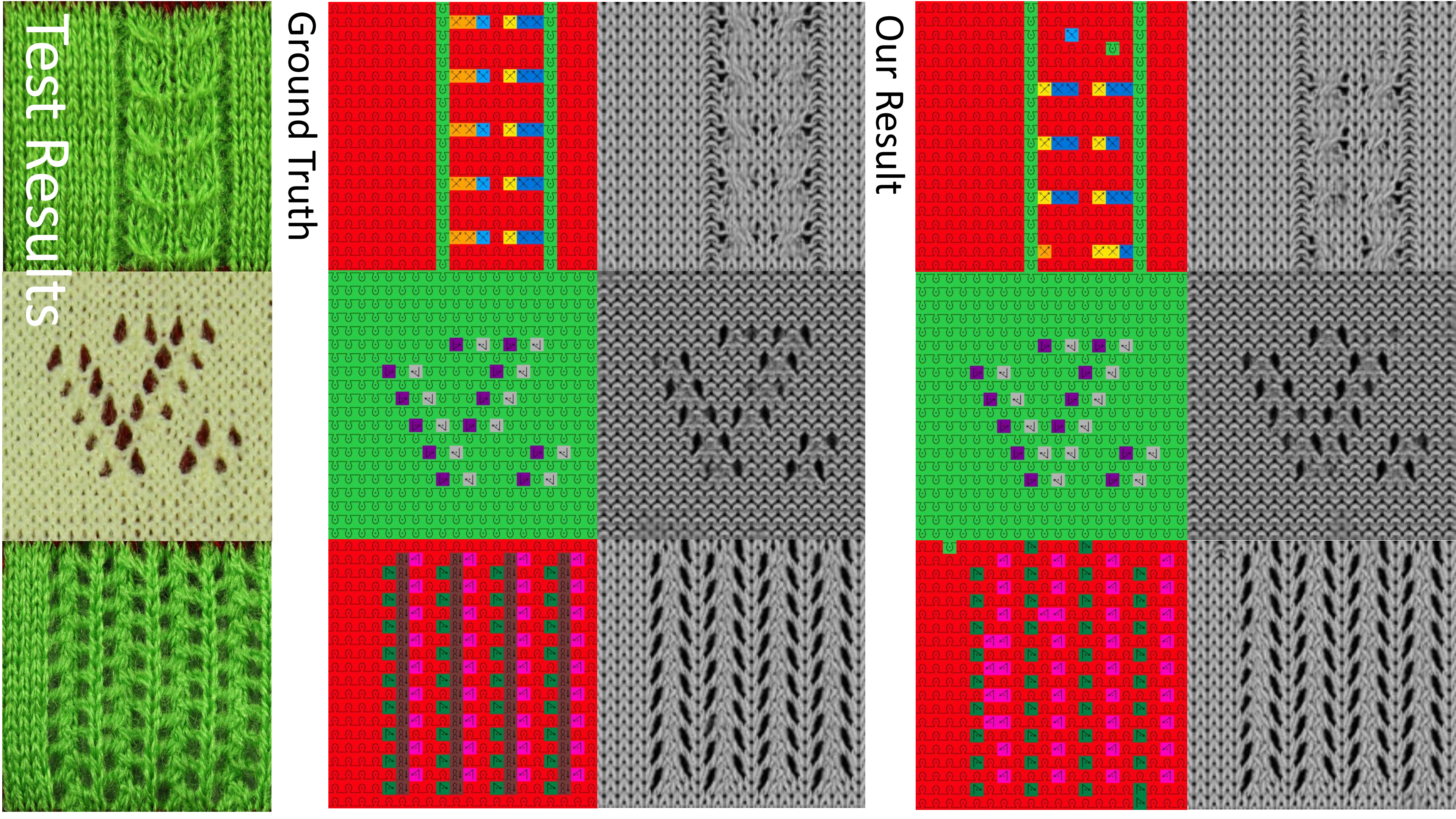


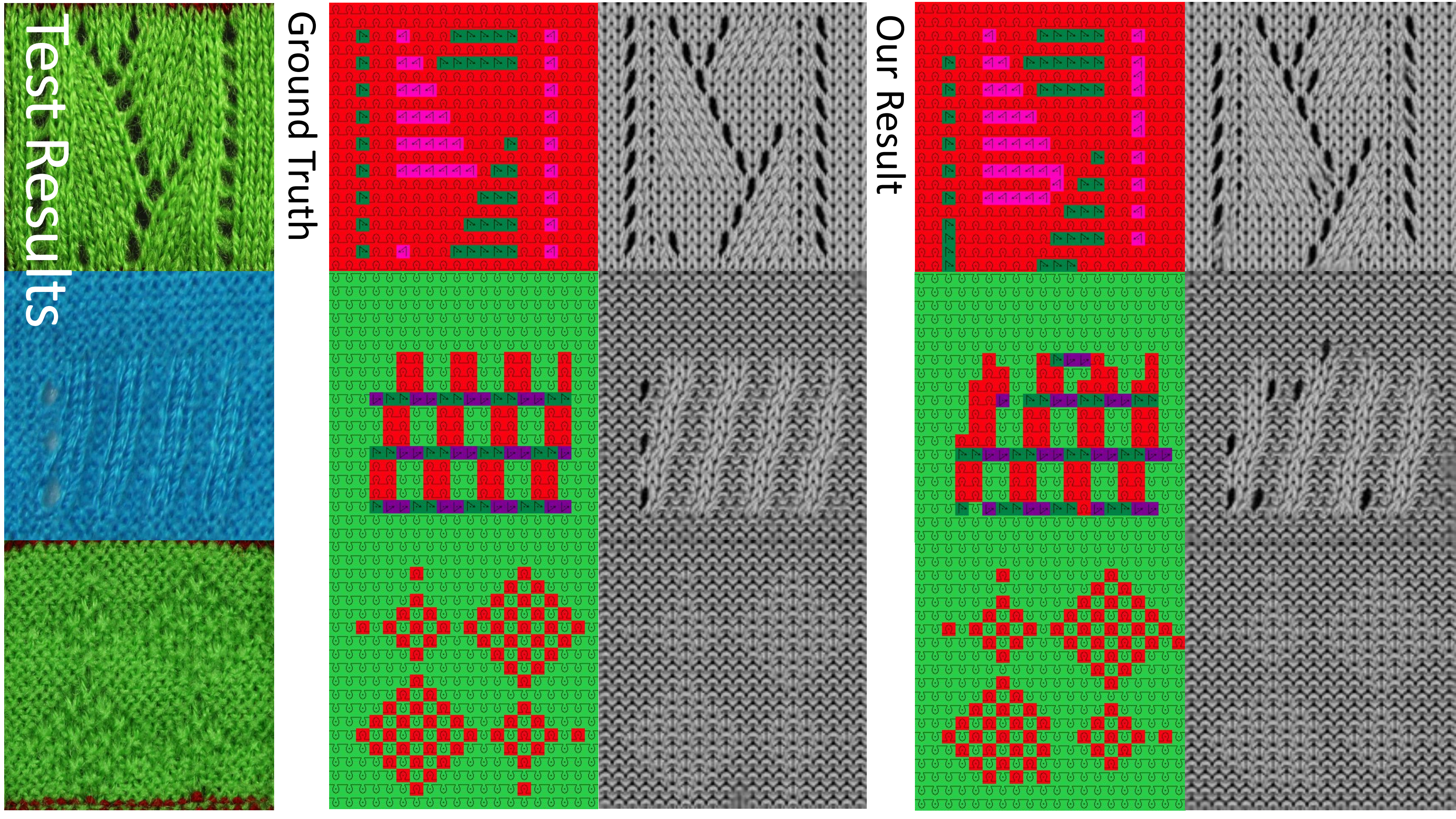
Network composition



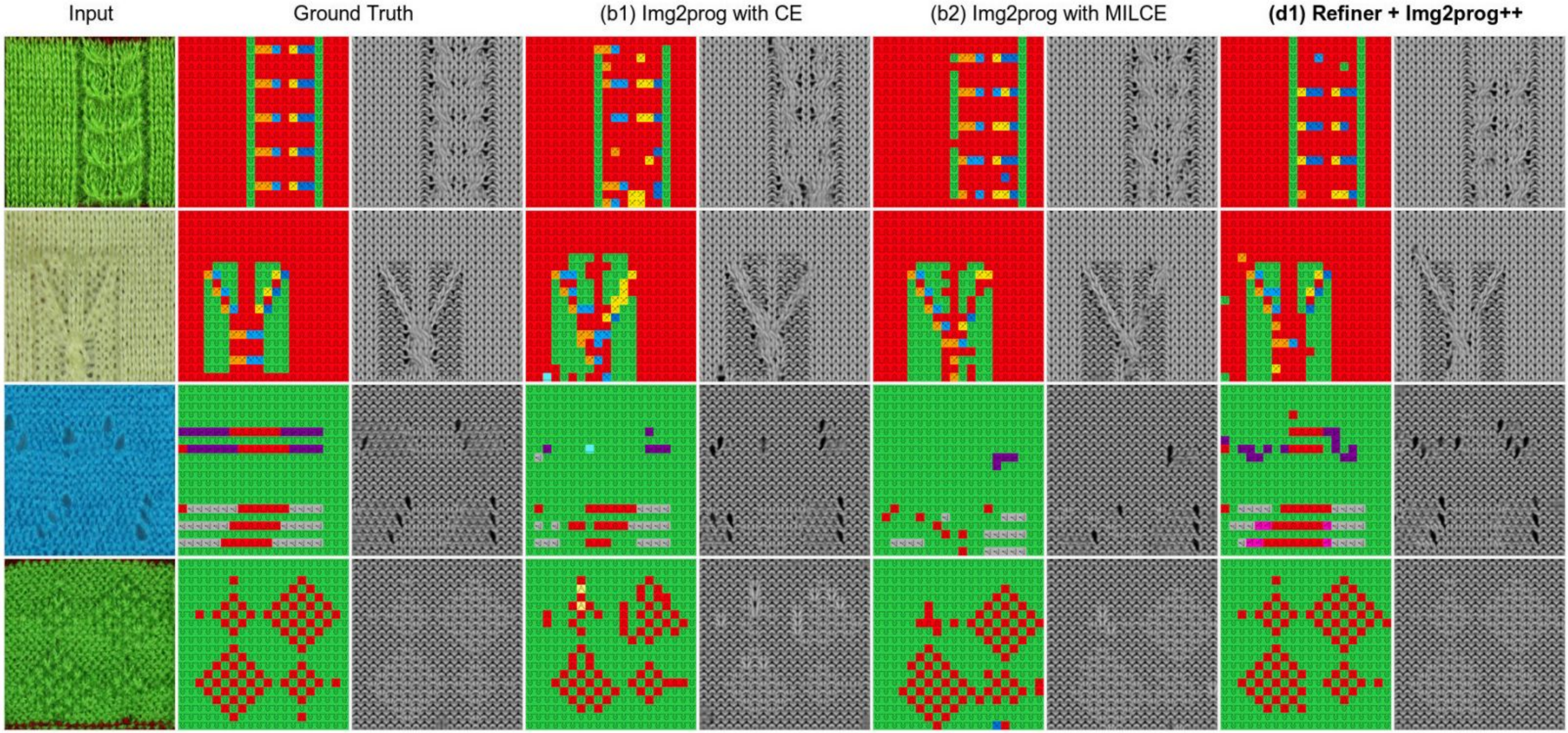
Results

Qualitative and quantitative evaluation


Test Results
Ground Truth
Our Result


Test Results
Ground Truth
Our Result

Architecture variations



Baseline, comparisons and impact of mixing

	Method	Accuracy (%)		Perceptual	
		Full	FG	SSIM	PSNR [dB]
(a1)	CycleGAN (Zhu et al., 2017)	57.27	24.10	0.670	15.87
(a2)	Pix2Pix (Isola et al., 2017)	56.20	47.98	0.660	15.95
(a3)	UNet (Ronneberger et al., 2015)	89.65	63.99	0.847	21.21
(a4)	Scene Parsing (Zhou et al., 2018)	**91.58**	**73.95**	**0.876**	**22.64**
(a5)	S+U (Shrivastava et al., 2017)	91.32	71.00	0.864	21.42
(b1)	`Img2prog` (real only) with CE	91.57	71.37	0.866	**21.62**
(b2)	`Img2prog` (real only) with MILCE	**91.74**	**72.30**	**0.871**	21.58
(c1)	Refiner + `Img2prog` ($\alpha = 0.9$)	93.48	78.53	0.894	**23.28**
(c2)	Refiner + `Img2prog` ($\alpha = 2/3$)	**93.58**	**78.57**	0.892	23.27
(c3)	Refiner + `Img2prog` ($\alpha = 0.5$)	93.57	78.30	**0.895**	23.24
(c4)	Refiner + `Img2prog` ($\alpha = 1/3$)	93.19	77.80	0.888	22.72
(c5)	Refiner + `Img2prog` ($\alpha = 0.1$)	92.42	74.15	0.881	22.27
(d1)	Refiner + `Img2prog++` ($\alpha = 0.5$)	**94.01**	**80.30**	**0.899**	**23.56**

Baseline, comparisons and impact of mixing

	Method	Accuracy (%)		Perceptual	
		Full	FG	SSIM	PSNR [dB]
(a1)	CycleGAN (Zhu et al., 2017)	57.27	24.10	0.670	15.87
(a2)	Pix2Pix (Isola et al., 2017)	56.20	47.98	0.660	15.95
(a3)	UNet (Ronneberger et al., 2015)	89.65	63.99	0.847	21.21
(a4)	Scene Parsing (Zhou et al., 2018)	**91.58**	**73.95**	**0.876**	**22.64**
(a5)	S+U (Shrivastava et al., 2017)	91.32	71.00	0.864	21.42
(b1)	`Img2prog` (real only) with CE	91.57	71.37	0.866	**21.62**
(b2)	`Img2prog` (real only) with MILCE	**91.74**	**72.30**	**0.871**	21.58
(c1)	Refiner + `Img2prog` ($\alpha = 0.9$)	93.48	78.53	0.894	**23.28**
(c2)	Refiner + `Img2prog` ($\alpha = 2/3$)	**93.58**	**78.57**	0.892	23.27
(c3)	Refiner + `Img2prog` ($\alpha = 0.5$)	93.57	78.30	**0.895**	23.24
(c4)	Refiner + `Img2prog` ($\alpha = 1/3$)	93.19	77.80	0.888	22.72
(c5)	Refiner + `Img2prog` ($\alpha = 0.1$)	92.42	74.15	0.881	22.27
(d1)	Refiner + `Img2prog++` ($\alpha = 0.5$)	**94.01**	**80.30**	**0.899**	**23.56**

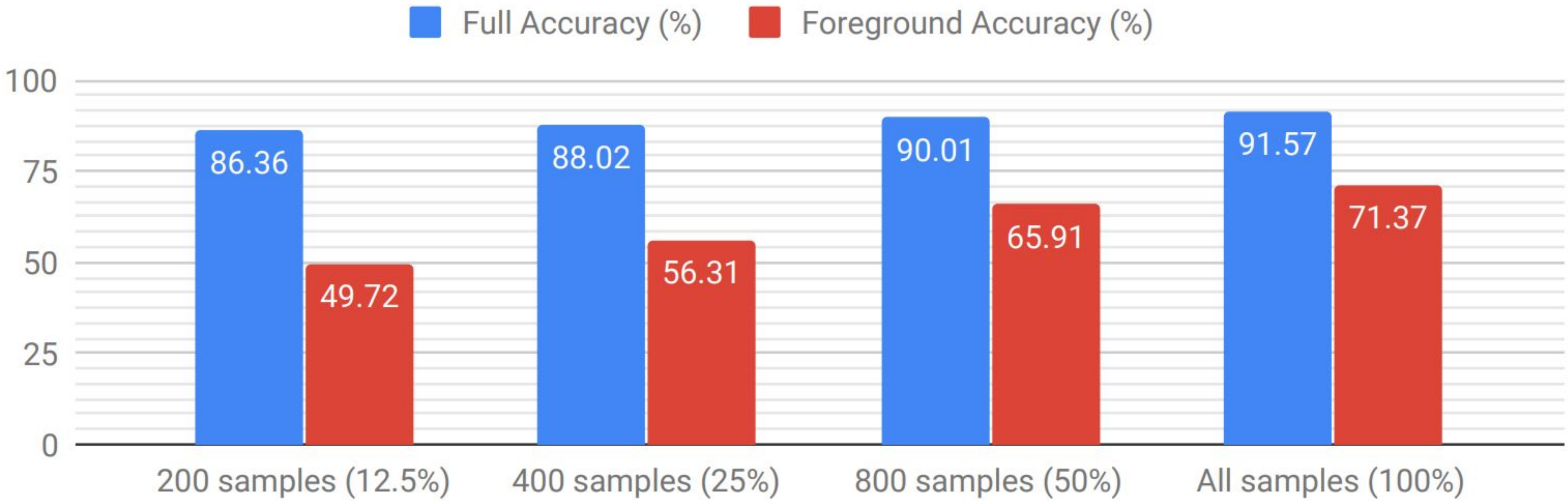
Baseline, comparisons and impact of mixing

	Method	Accuracy (%)		Perceptual	
		Full	FG	SSIM	PSNR [dB]
(a1)	CycleGAN (Zhu et al., 2017)	57.27	24.10	0.670	15.87
(a2)	Pix2Pix (Isola et al., 2017)	56.20	47.98	0.660	15.95
(a3)	UNet (Ronneberger et al., 2015)	89.65	63.99	0.847	21.21
(a4)	Scene Parsing (Zhou et al., 2018)	**91.58**	**73.95**	**0.876**	**22.64**
(a5)	S+U (Shrivastava et al., 2017)	91.32	71.00	0.864	21.42
(b1)	`Img2prog` (real only) with CE	91.57	71.37	0.866	**21.62**
(b2)	`Img2prog` (real only) with MILCE	**91.74**	**72.30**	**0.871**	21.58
(c1)	Refiner + `Img2prog` ($\alpha = 0.9$)	93.48	78.53	0.894	**23.28**
(c2)	Refiner + `Img2prog` ($\alpha = 2/3$)	**93.58**	**78.57**	0.892	23.27
(c3)	Refiner + `Img2prog` ($\alpha = 0.5$)	93.57	78.30	**0.895**	23.24
(c4)	Refiner + `Img2prog` ($\alpha = 1/3$)	93.19	77.80	0.888	22.72
(c5)	Refiner + `Img2prog` ($\alpha = 0.1$)	92.42	74.15	0.881	22.27
(d1)	Refiner + `Img2prog++` ($\alpha = 0.5$)	**94.01**	**80.30**	**0.899**	**23.56**

Baseline, comparisons and impact of mixing

	Method	Accuracy (%)		Perceptual	
		Full	FG	SSIM	PSNR [dB]
(a1)	CycleGAN (Zhu et al., 2017)	57.27	24.10	0.670	15.87
(a2)	Pix2Pix (Isola et al., 2017)	56.20	47.98	0.660	15.95
(a3)	UNet (Ronneberger et al., 2015)	89.65	63.99	0.847	21.21
(a4)	Scene Parsing (Zhou et al., 2018)	**91.58**	**73.95**	**0.876**	**22.64**
(a5)	S+U (Shrivastava et al., 2017)	91.32	71.00	0.864	21.42
(b1)	`Img2prog` (real only) with CE	91.57	71.37	0.866	**21.62**
(b2)	`Img2prog` (real only) with MILCE	**91.74**	**72.30**	**0.871**	21.58
(c1)	Refiner + `Img2prog` ($\alpha = 0.9$)	93.48	78.53	0.894	**23.28**
(c2)	Refiner + `Img2prog` ($\alpha = 2/3$)	**93.58**	**78.57**	0.892	23.27
(c3)	Refiner + `Img2prog` ($\alpha = 0.5$)	93.57	78.30	**0.895**	23.24
(c4)	Refiner + `Img2prog` ($\alpha = 1/3$)	93.19	77.80	0.888	22.72
(c5)	Refiner + `Img2prog` ($\alpha = 0.1$)	92.42	74.15	0.881	22.27
(d1)	Refiner + `Img2prog++` ($\alpha = 0.5$)	**94.01**	**80.30**	**0.899**	**23.56**

How much data is enough data?



Limitations

And potential solutions

Issue of scale, stretch and orientation

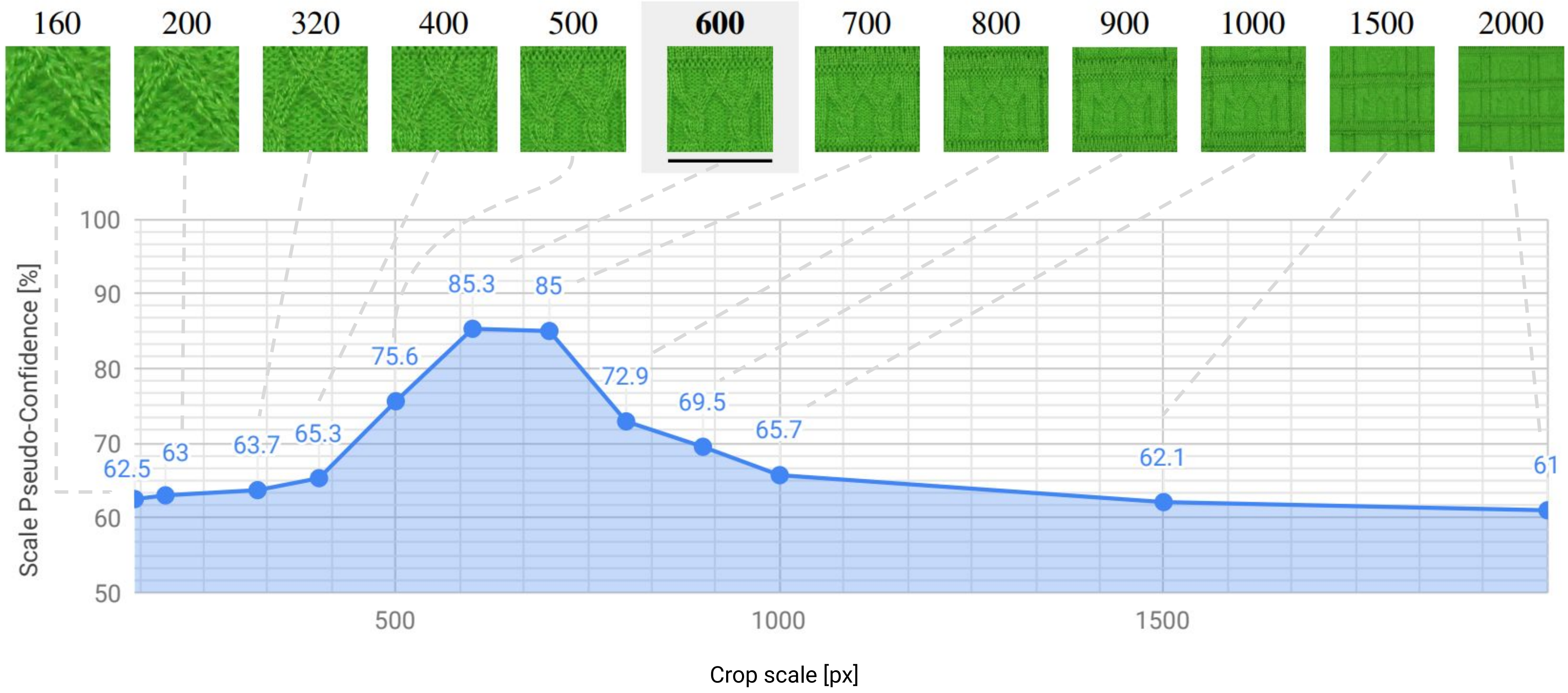
We assume a specific scale, stretch (of 20x20 stitches) and a bottom-up orientation of stitch courses.

Options:

- Explicit model scale, stretch and orientation
  = makes training more complicated

- Separate selection (using measure of “confidence”)
  = take large-scale image, and try space of scales / stretches / rot.

Attempt at scale selection (successful)



Input variety

We only used Tamm 2/30 acrylic yarn.

How do we scale to more data, and more varieties of it?

Options:

- **Simulation**: need fast simulation of yarn (hard, or slow), hopefully as a differentiable renderer (within the network)
- **Online yarn images**: unsupervised way? Cycle-consistency? Additional side/weaker/stronger task?

Modeling Hard Constraints

Currently, output may have invalid instruction combinations.

Tried to use penalty on valid 1st order neighborhood, but little impact.

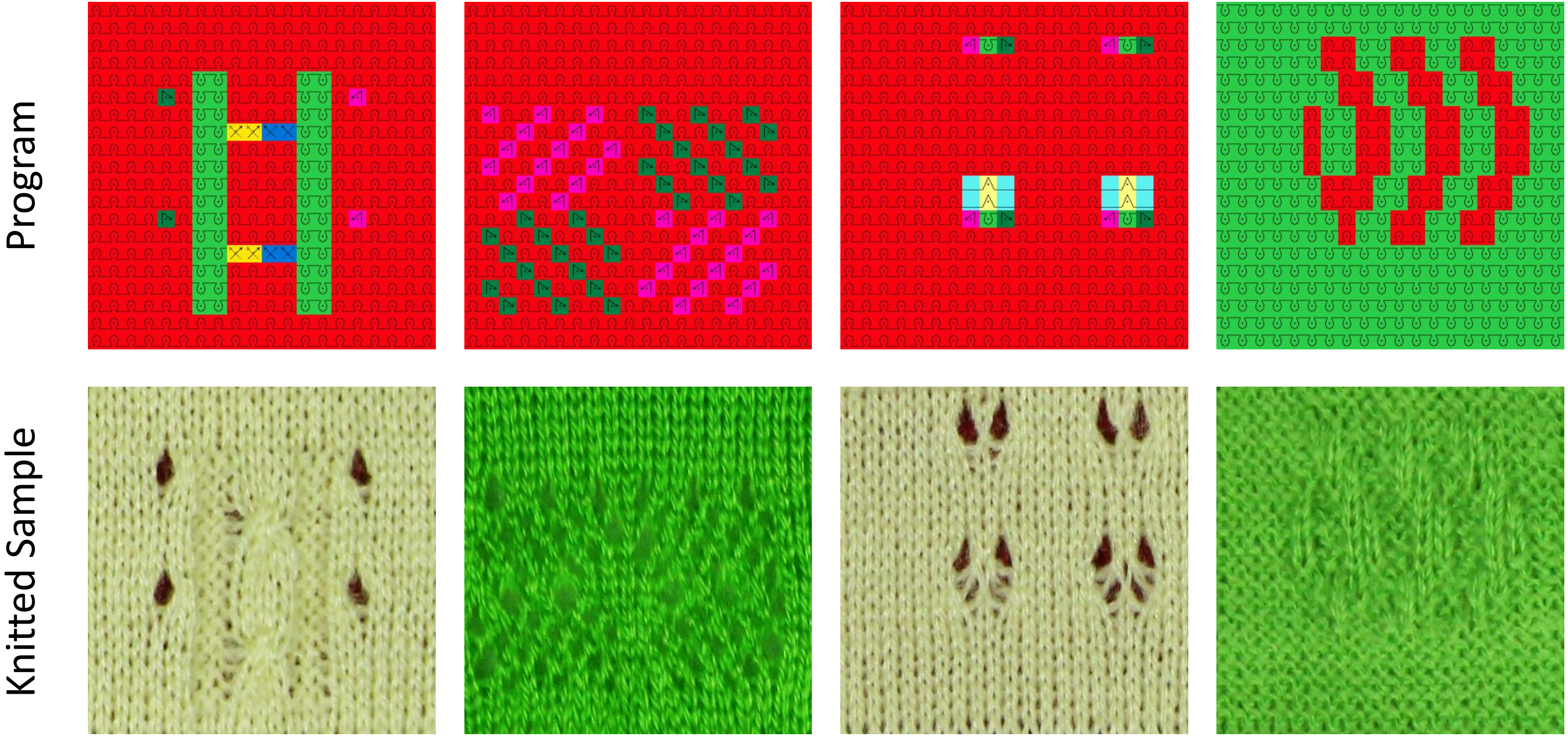
Questions:

- How do we model hard constraints with a neural network?
- Split translation into instruction "potentials" and then select the actual instructions (e.g., using known knittability constraints)?
- Can we infer the syntax constraints automatically?
  - Note: non-trivial to specify beyond first-order neighborhood unless enough data is available…

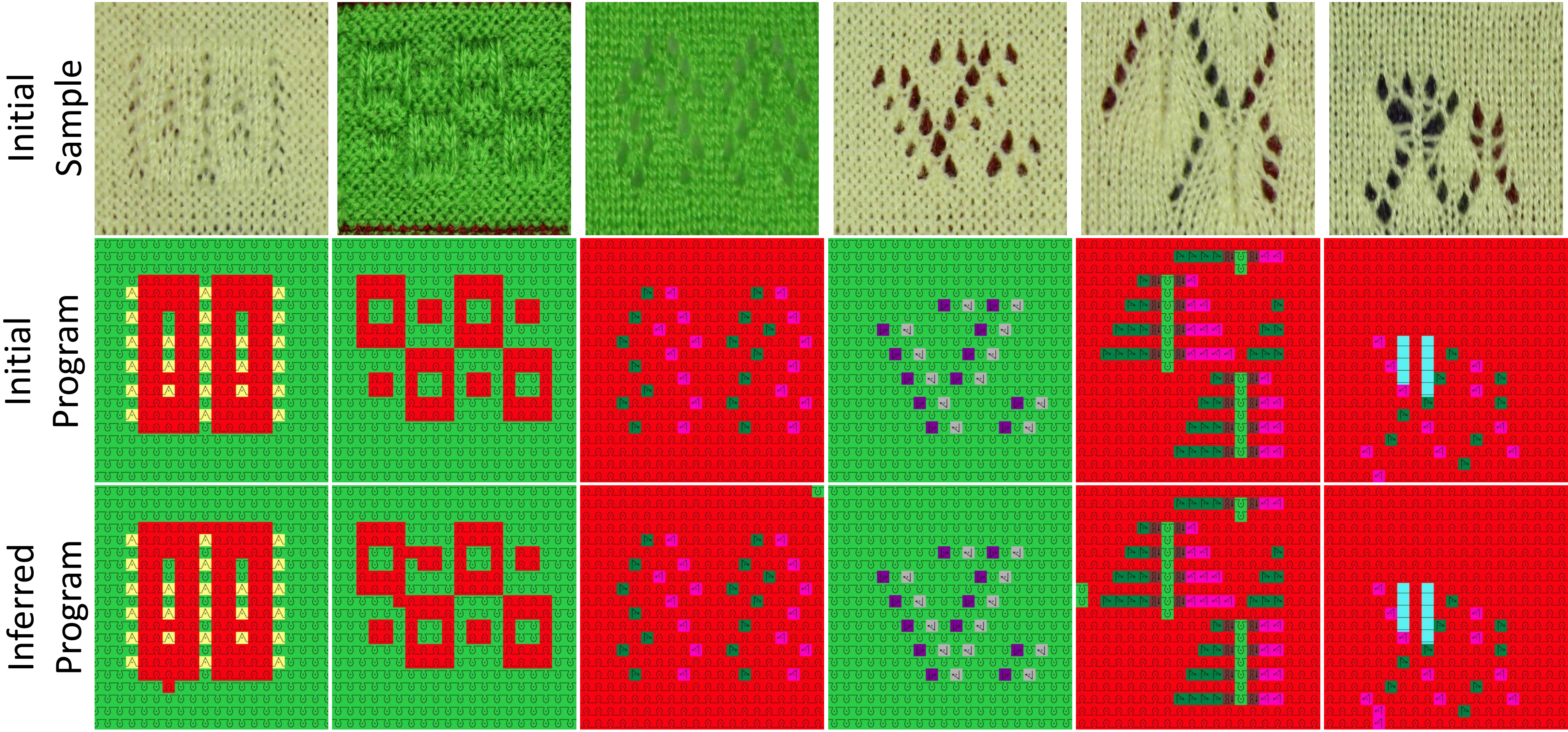
Result Details

The great, the good, the not so good, and the ugly

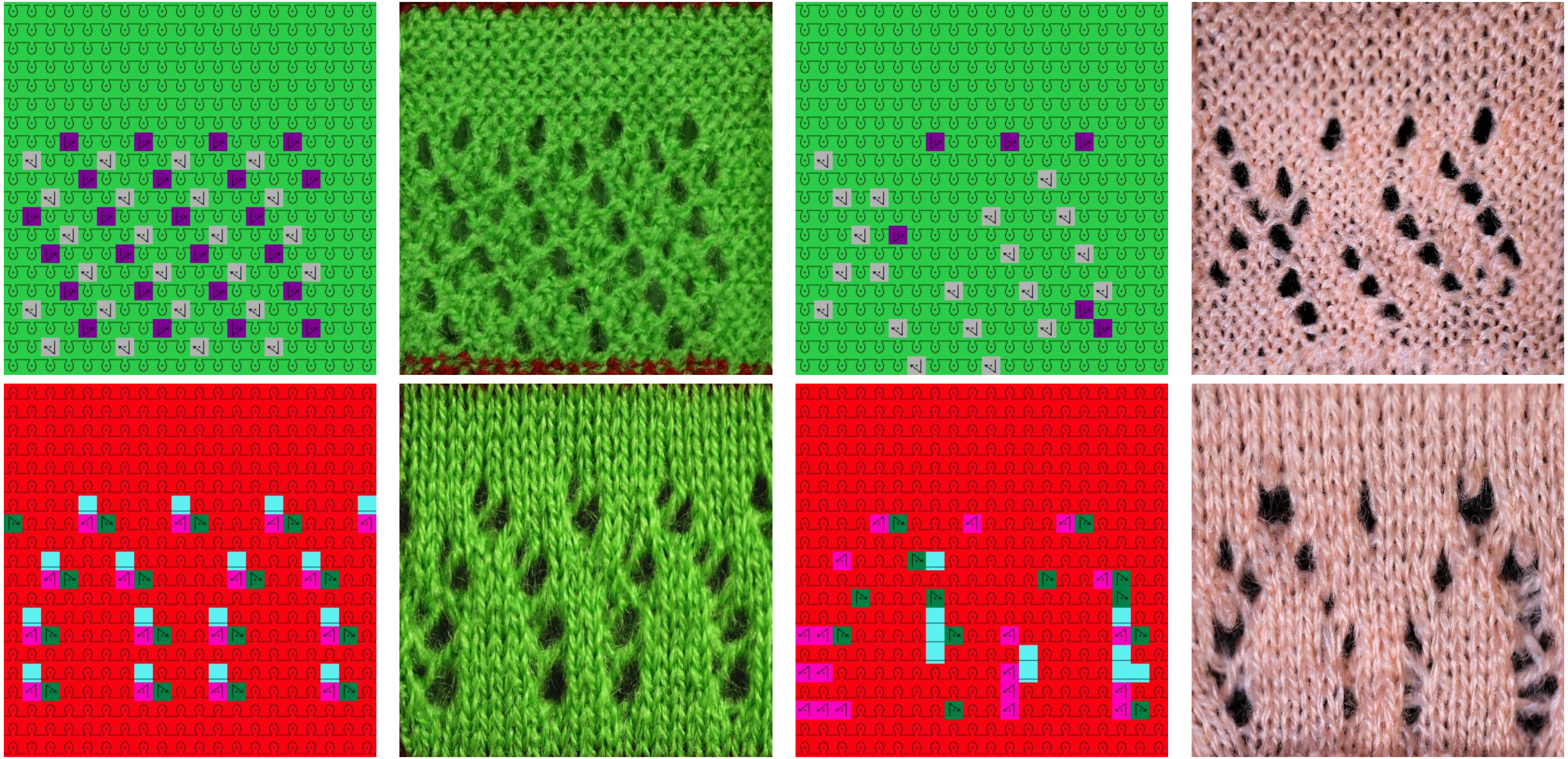
Details: Perfect cases



Details: Minor errors (no semantic issue)

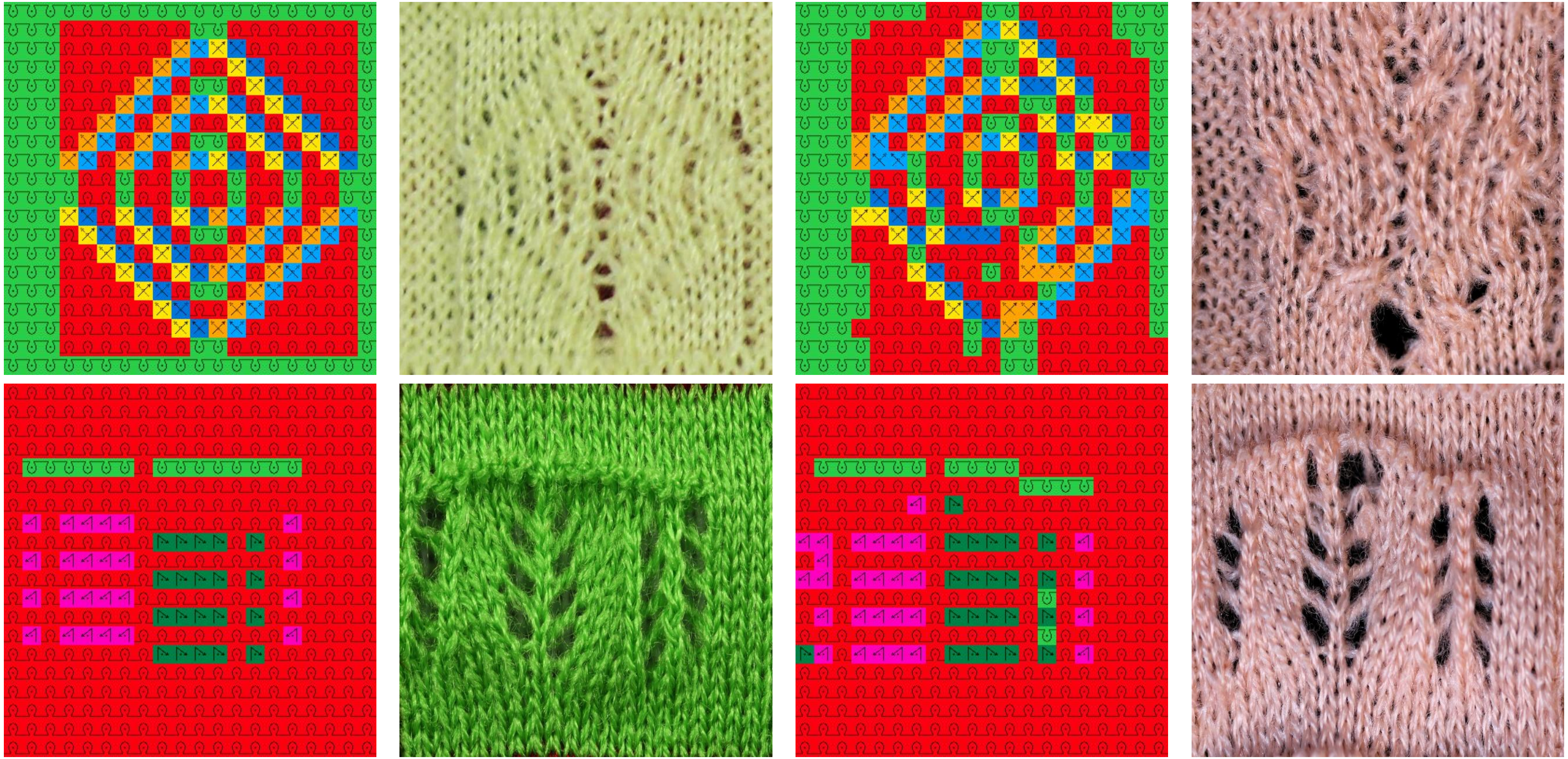


Details: Larger errors (but knittable)



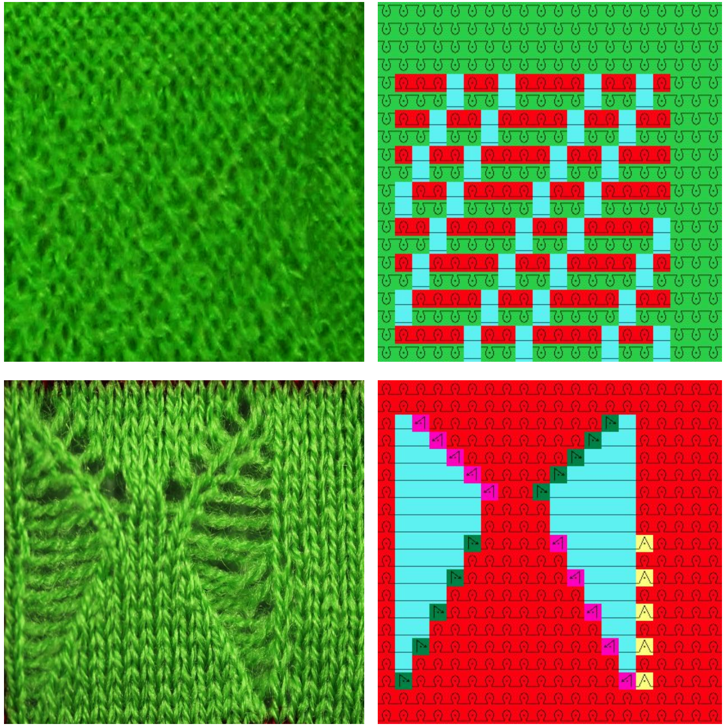
Initial program | Initial sample | Inferred program | Final sample

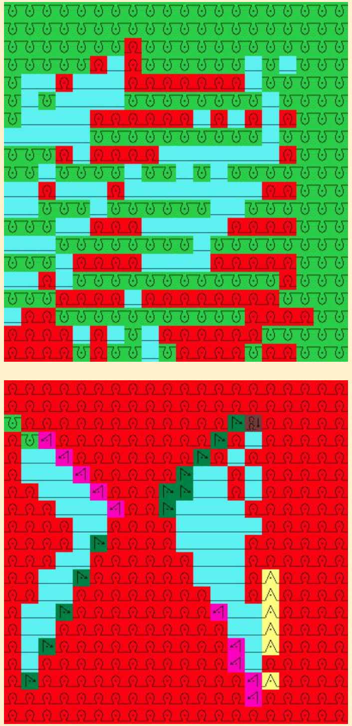
Details: Larger errors (but knittable)



Initial program | Initial sample | Inferred program | Final sample

Details: (few) catastrophic failures (only 2)





Not knittable!

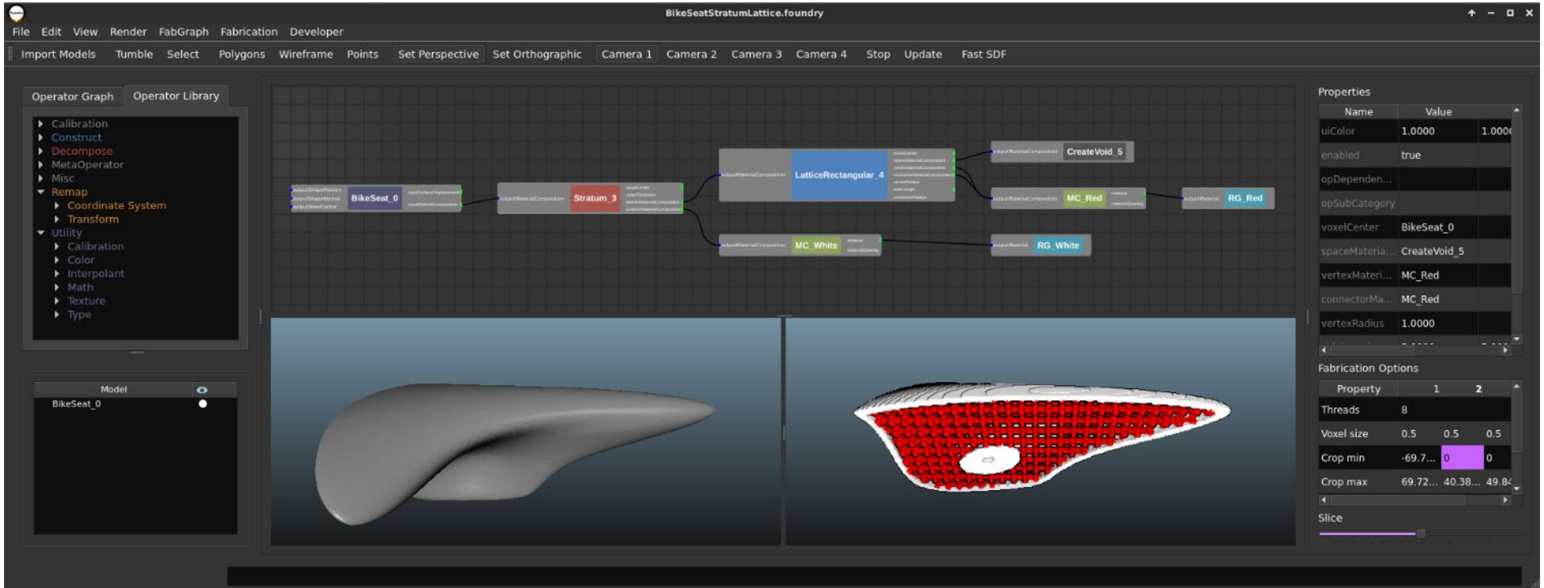
/!\

Yarn collapses!

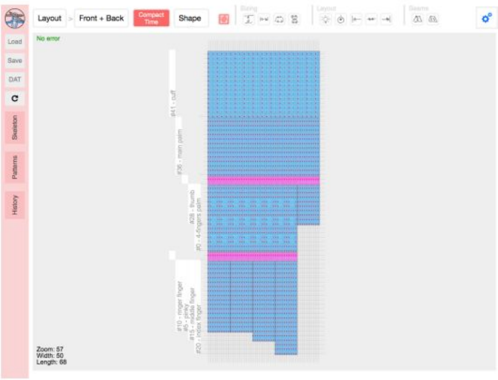
Past and Future Work

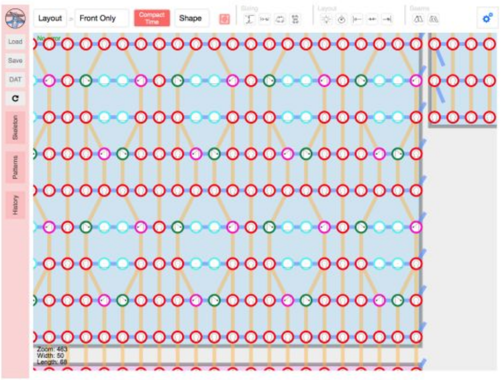
Where it came from, and where it is going

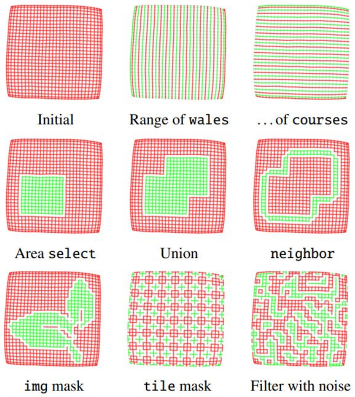
Past: Foundry - Multi-Material 3D Printing

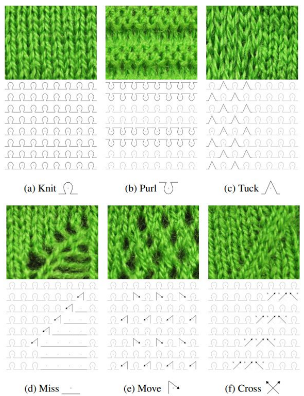


Recent: Knitting Skeletons - CAD for Knitting









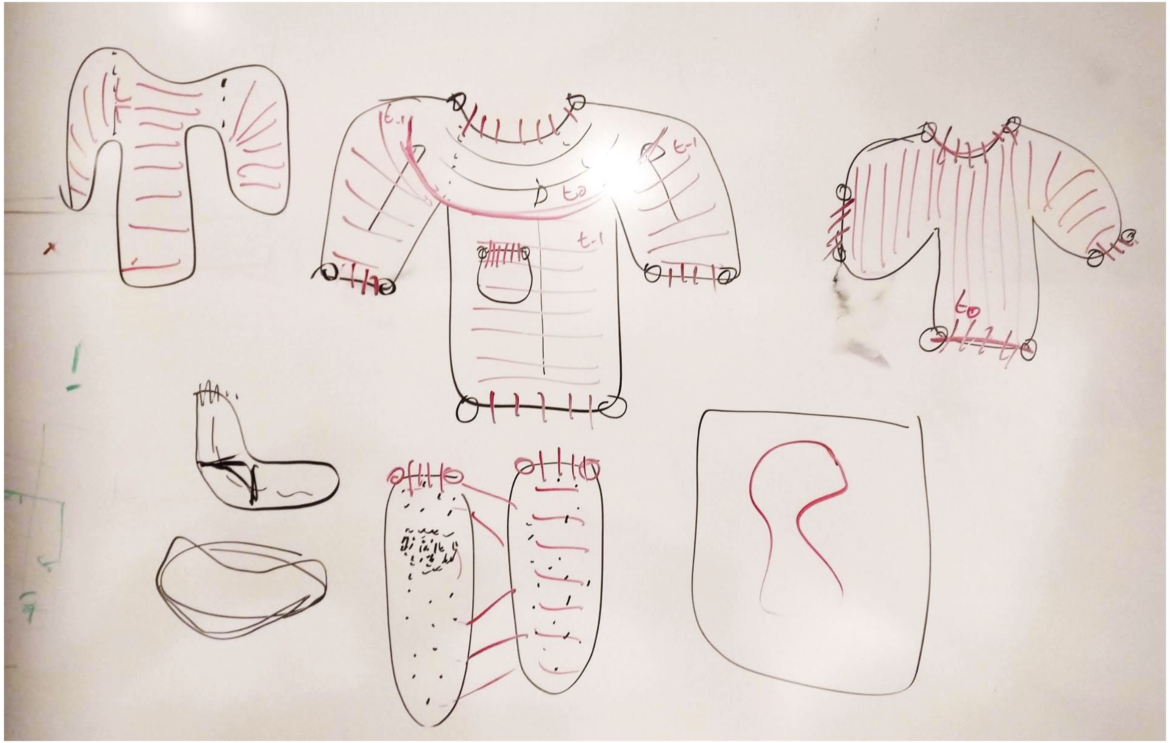
Now: Knit Sketching - Sketches within CAD

Work with sketches

- Wale flow
- Connectivity
- Stitch density
- Layers (sketches)
- Layers (patterns)

Generate data for the CAD system.

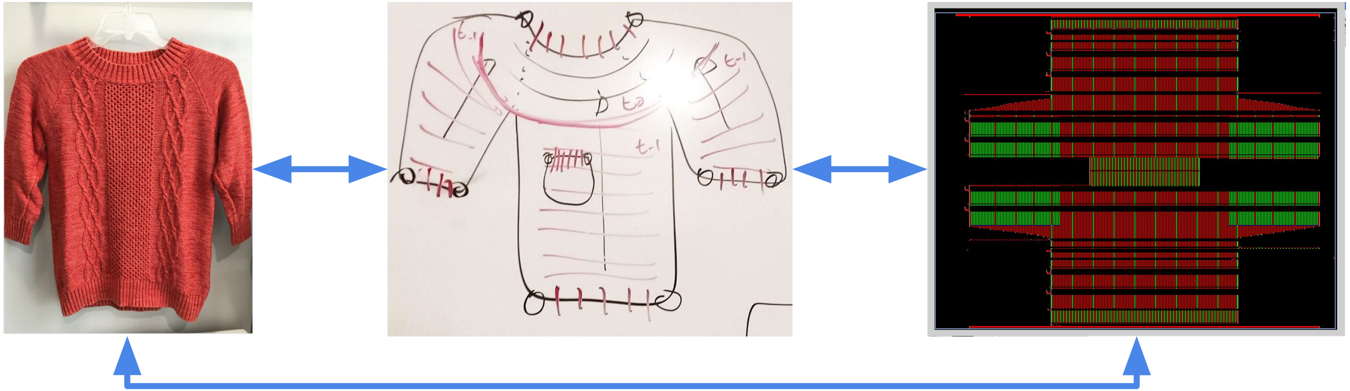
(with some efficient parameterization)



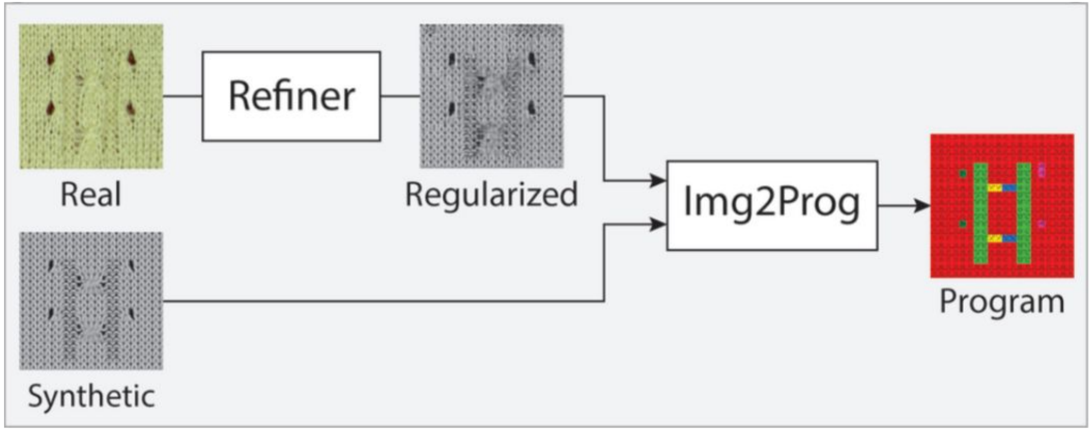
Next: InverseKnit++

Use sketch input capability to learn to map full knitted "shapes" directly into low-level knitting programs.

- More instruction irregularities
- Issue of occlusion (two-sided shapes)
- Ambiguity between shape and patterns



http://deepknitting.csail.mit.edu



Thank you for listening!

Dataset Details

Instruction distribution and accuracies

Dataset: instruction statistics

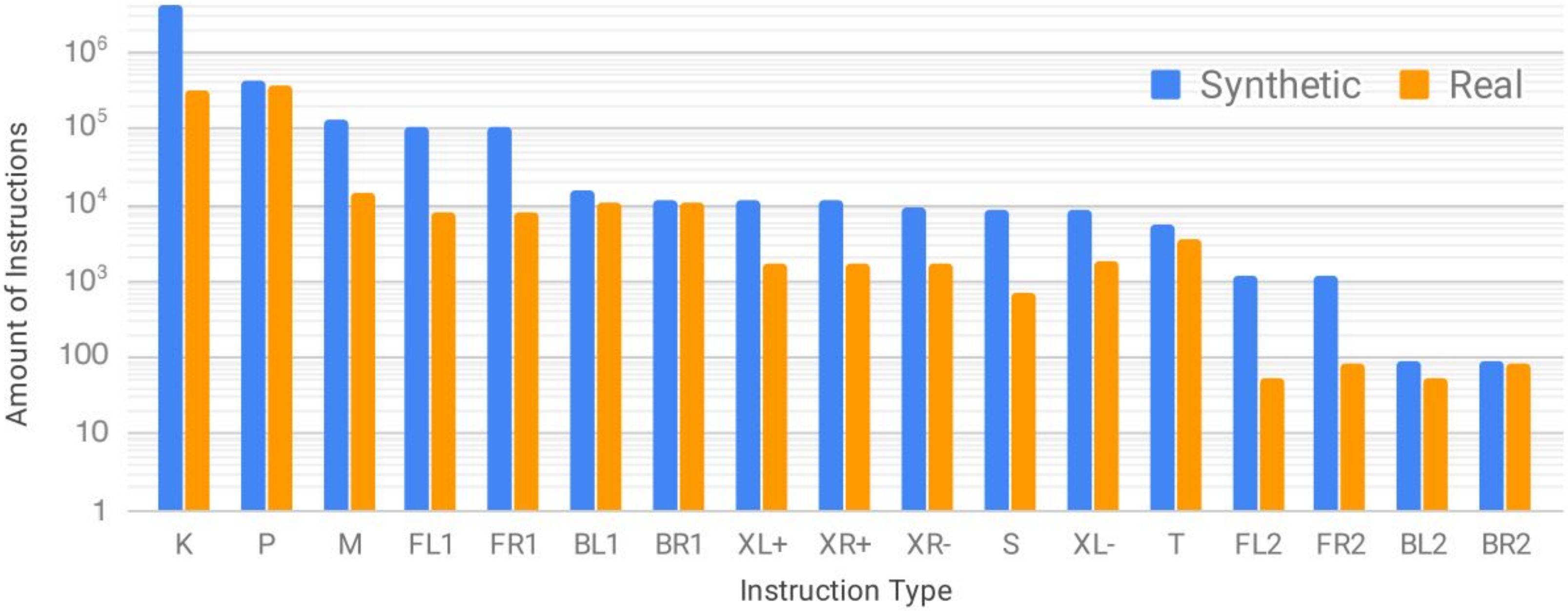


Figure 5. Instruction counts in descending order, for synthetic and real images. Note the logarithmic scale of the Y axis.

Per-Instruction Accuracies

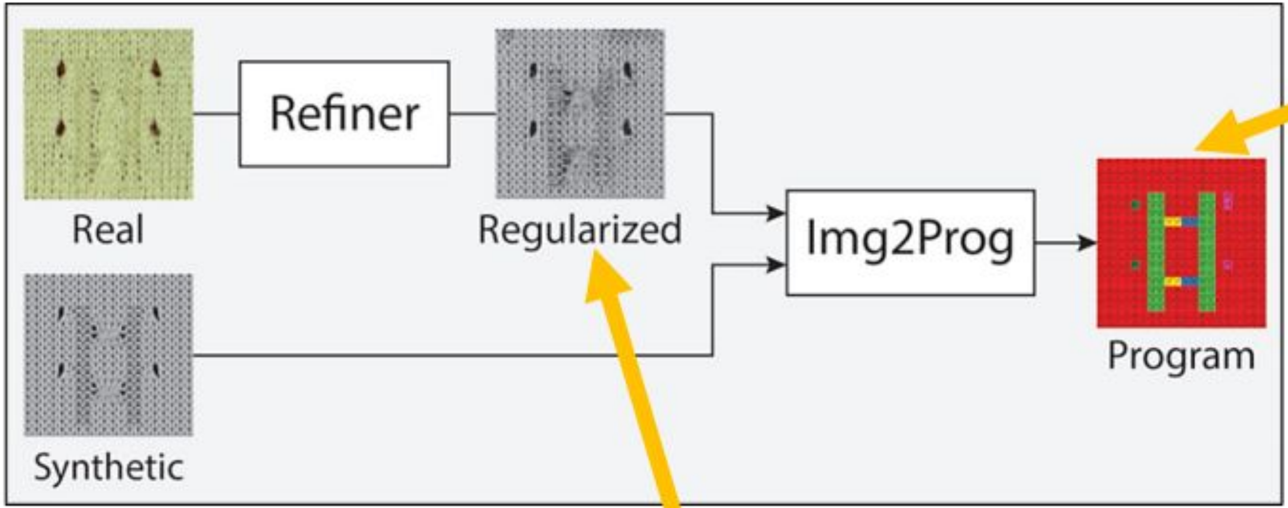
Table 2. **Performance of *Refined+Img2prog++* measured per instruction over the test set.** This shows that even though our instruction distribution has very large variations, our network is still capable of learning some representation for the least frequent instructions (3 orders of magnitude difference for FR2, FL2, BR2, BL2 compared to K and P).

Instruction	K	P	T	M	FR1	FR2	FL1	FL2	BR1	BR2	BL1	BL2	XR+	XR-	XL+	XL-	S
Accuracy [%]	96.52	96.64	74.63	66.65	77.16	100.00	74.20	83.33	68.73	27.27	69.94	22.73	60.15	62.33	60.81	62.11	25.85
Frequency [%]	44.39	47.72	0.41	1.49	1.16	0.01	1.23	0.01	1.22	0.02	1.40	0.02	0.22	0.18	0.19	0.22	0.12

Architecture Details

Neural Networks and Losses

Actual Loss Function



2D cross entropy

Style loss + GAN loss

Actual Loss Function

Our combined loss is the weighted sum

$$\mathcal{L} = \lambda_{\text{CE}}\mathcal{L}_{\text{CE}} + \lambda_{\text{Perc}}\mathcal{L}_{\text{Perc}} + \lambda_{\text{GAN}}\mathcal{L}_{\text{GAN}} \tag{5}$$

where we used the weights: $\lambda_{\text{CE}} = 3$, $\lambda_{\text{Perc}} = 0.02/(128)^2$ and $\lambda_{\text{GAN}} = 0.2$. The losses $\mathcal{L}_{\text{Perc}}$ and λ_{GAN} are measured on the output of `Refiner`, while the loss λ_{CE} is measured on `Img2prog`.

Refiner Network

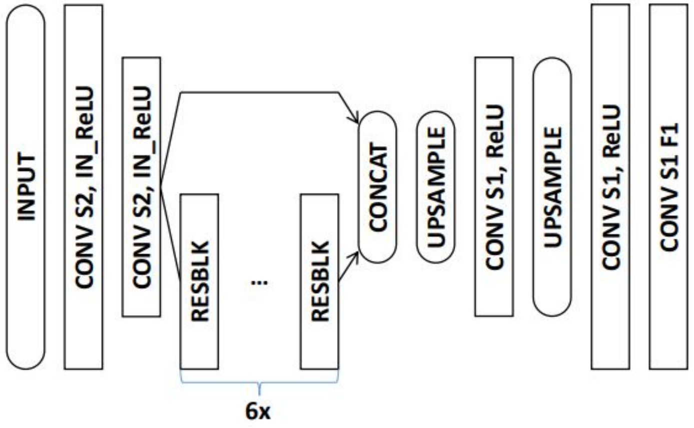


Figure 10. The illustration of the `Refiner` network architecture, where S$\#N$ denotes the stride size of $\#N$, `IN_ReLU` indicates the Instance normalization followed by ReLU, `Resblk` is the residual block that consists of ConvS1-ReLU-ConvS1 with short-cut connection (He et al., 2016), Upsample is the nearest neighbor upsampling with the factor $2\times$, F is the output channel dimension. If not mentioned, the default parameters for all the convolutions are the stride size of 2, $F = 64$, and the 3×3 kernel size.

Theorem 1

About the Generalization Gap

Definition 1: Discrepancy [Mansour 09]

Definition 1 (Discrepancy (Mansour et al., 2009)). *Let $\mathcal{H}$ be a class of functions mapping from $\mathcal{X}$ to $\mathcal{Y}$. The discrepancy between two distribution $\mathcal{D}_1$ and $\mathcal{D}_2$ over $\mathcal{X}$ is defined as*

$$\mathrm{disc}_{\mathcal{H}}(\mathcal{D}_1, \mathcal{D}_2) = \max_{h,h' \in \mathcal{H}} \left|\mathcal{L}_{\mathcal{D}_1}(h, h') - \mathcal{L}_{\mathcal{D}_2}(h, h')\right|. \quad (6)$$

The discrepancy is symmetric and satisfies the triangle inequality, regardless of any loss function. This can be used to compare distributions for general tasks even including regression.

Lemma 1: from Lemma 4 of [Ben-David 10]

Lemma 1. *Let h be a hypothesis in class $\mathcal{H}$, and assume that $\mathcal{L}$ is symmetric and obeys the triangle inequality. Then*

$$|\mathcal{L}_\alpha(h, y) - \mathcal{L}_T(h, y)| \leq \alpha \left(\mathrm{disc}_{\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T) + \lambda \right), \quad (7)$$

where $\lambda = \mathcal{L}_S(h^*, y) + \mathcal{L}_T(h^*, y)$, and the ideal joint hypothesis h^* is defined as $h^* = \arg\min_{h \in \mathcal{H}} \mathcal{L}_S(h, y) + \mathcal{L}_T(h, y)$.

Lemma 1: from Lemma 4 of [Ben-David 10]

Proof. The proof is based on the triangle inequality of $\mathcal{L}$, and the last inequality follows the definition of the discrepancy.

Substitute $\mathcal{L}_\alpha = \alpha\, \mathcal{L}_S + (1 - \alpha)\, \mathcal{L}_T$

$$
\begin{aligned}
&|\mathcal{L}_\alpha(h,y) - \mathcal{L}_T(h,y)| \\
=&\alpha|\mathcal{L}_S(h,y) - \mathcal{L}_T(h,y)| \\
=&\alpha\,|\mathcal{L}_S(h,y) - \mathcal{L}_S(h^*,h) + \mathcal{L}_S(h^*,h) \\
&\quad - \mathcal{L}_T(h^*,h) + \mathcal{L}_T(h^*,h) - \mathcal{L}_T(h,y)\,| \\
\leq&\alpha\big|\,|\mathcal{L}_S(h,y) - \mathcal{L}_S(h^*,h)| + \\
&\quad |\mathcal{L}_S(h^*,h) - \mathcal{L}_T(h^*,h)| + |\mathcal{L}_T(h^*,h) - \mathcal{L}_T(h,y)|\,\big| \\
\leq&\alpha\big|\mathcal{L}_S(h^*,y) + |\mathcal{L}_S(h^*,h) - \mathcal{L}_T(h^*,h)| + \mathcal{L}_T(h^*,y)\big| \\
\leq&\alpha\,(\mathrm{disc}_{\mathcal{H}}(\mathcal{D}_S,\mathcal{D}_T) + \lambda)\,. \qquad (8)
\end{aligned}
$$

Lemma 1: from Lemma 4 of [Ben-David 10]

Proof. The proof is based on the triangle inequality of $\mathcal{L}$, and the last inequality follows the definition of the discrepancy.

$$
\begin{aligned}
& |\mathcal{L}_\alpha(h, y) - \mathcal{L}_T(h, y)| \\
=& \alpha|\mathcal{L}_S(h, y) - \mathcal{L}_T(h, y)| \\
=& \alpha\,|\mathcal{L}_S(h, y) - \mathcal{L}_S(h^*, h) + \mathcal{L}_S(h^*, h) \\
& - \mathcal{L}_T(h^*, h) + \mathcal{L}_T(h^*, h) - \mathcal{L}_T(h, y)\,| \\
\le& \alpha\big|\,|\mathcal{L}_S(h, y) - \mathcal{L}_S(h^*, h)| + \\
& |\mathcal{L}_S(h^*, h) - \mathcal{L}_T(h^*, h)| + |\mathcal{L}_T(h^*, h) - \mathcal{L}_T(h, y)|\,\big| \\
\le& \alpha\big|\mathcal{L}_S(h^*, y) + |\mathcal{L}_S(h^*, h) - \mathcal{L}_T(h^*, h)| + \mathcal{L}_T(h^*, y)\big| \\
\le& \alpha\,(\mathrm{disc}_{\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T) + \lambda)\,. \qquad (8)
\end{aligned}
$$

Introduce $(\mathcal{L}_S(h^*,h) - \mathcal{L}_S(h^*,h))$ and $(\mathcal{L}_T(h^*,h)-\mathcal{L}_T(h^*,h))$

Lemma 1: from Lemma 4 of [Ben-David 10]

Proof. The proof is based on the triangle inequality of $\mathcal{L}$, and the last inequality follows the definition of the discrepancy.

$$
\begin{aligned}
&|\mathcal{L}_\alpha(h,y) - \mathcal{L}_T(h,y)| \\
=&\alpha|\mathcal{L}_S(h,y) - \mathcal{L}_T(h,y)| \\
=&\alpha\,|\mathcal{L}_S(h,y) - \mathcal{L}_S(h^*,h) + \mathcal{L}_S(h^*,h) \\
&- \mathcal{L}_T(h^*,h) + \mathcal{L}_T(h^*,h) - \mathcal{L}_T(h,y)\,| \\
\leq&\alpha\big|\,|\mathcal{L}_S(h,y) - \mathcal{L}_S(h^*,h)| + \\
&|\mathcal{L}_S(h^*,h) - \mathcal{L}_T(h^*,h)| + |\mathcal{L}_T(h^*,h) - \mathcal{L}_T(h,y)|\,\big| \\
\leq&\alpha\big|\mathcal{L}_S(h^*,y)+|\mathcal{L}_S(h^*,h)-\mathcal{L}_T(h^*,h)|+\mathcal{L}_T(h^*,y)\big| \\
\leq&\alpha\,(\mathrm{disc}_{\mathcal{H}}(\mathcal{D}_S,\mathcal{D}_T) + \lambda)\,. \qquad (8)
\end{aligned}
$$

Apply triangular inequality

Lemma 1: from Lemma 4 of [Ben-David 10]

Proof. The proof is based on the triangle inequality of $\mathcal{L}$, and the last inequality follows the definition of the discrepancy.

$$
\begin{aligned}
&|\mathcal{L}_\alpha(h,y) - \mathcal{L}_T(h,y)| \\
=&\alpha|\mathcal{L}_S(h,y) - \mathcal{L}_T(h,y)| \\
=&\alpha\,|\mathcal{L}_S(h,y) - \mathcal{L}_S(h^*,h) + \mathcal{L}_S(h^*,h) \\
&\quad - \mathcal{L}_T(h^*,h) + \mathcal{L}_T(h^*,h) - \mathcal{L}_T(h,y)\,| \\
\leq&\alpha\big|\boxed{|\mathcal{L}_S(h,y) - \mathcal{L}_S(h^*,h)|}+ \\
&\quad \boxed{|\mathcal{L}_S(h^*,h) - \mathcal{L}_T(h^*,h)|}+\boxed{|\mathcal{L}_T(h^*,h) - \mathcal{L}_T(h,y)|}\big| \\
\leq&\alpha\big|\boxed{\mathcal{L}_S(h^*,y)}+\boxed{|\mathcal{L}_S(h^*,h)-\mathcal{L}_T(h^*,h)|}+\boxed{\mathcal{L}_T(h^*,y)} \\
\leq&\alpha\,(\mathrm{disc}_{\mathcal{H}}(\mathcal{D}_S,\mathcal{D}_T) + \lambda)\,. \qquad (8)
\end{aligned}
$$

Apply triangular inequality
AB <= AC + BC
AB - AC <= BC

Lemma 1: from Lemma 4 of [Ben-David 10]

Proof. The proof is based on the triangle inequality of $\mathcal{L}$, and the last inequality follows the definition of the discrepancy.

$$
\begin{aligned}
&|\mathcal{L}_\alpha(h,y) - \mathcal{L}_T(h,y)| \\
=&\alpha|\mathcal{L}_S(h,y) - \mathcal{L}_T(h,y)| \\
=&\alpha\,|\mathcal{L}_S(h,y) - \mathcal{L}_S(h^*,h) + \mathcal{L}_S(h^*,h) \\
&- \mathcal{L}_T(h^*,h) + \mathcal{L}_T(h^*,h) - \mathcal{L}_T(h,y)\,| \\
\leq&\alpha\big|\,|\mathcal{L}_S(h,y) - \mathcal{L}_S(h^*,h)| + \\
&|\mathcal{L}_S(h^*,h) - \mathcal{L}_T(h^*,h)| + |\mathcal{L}_T(h^*,h) - \mathcal{L}_T(h,y)|\,\big| \\
\leq&\alpha\big|\mathcal{L}_S(h^*,y) + |\mathcal{L}_S(h^*,h) - \mathcal{L}_T(h^*,h)| + \mathcal{L}_T(h^*,y)\big| \\
\leq&\alpha\,(\mathrm{disc}_{\mathcal{H}}(\mathcal{D}_S,\mathcal{D}_T) + \lambda)\,. \qquad (8)
\end{aligned}
$$

Definition of discrepancy
+ storing rest in λ

Lemma 2: from [Ben-David 10]

Lemma 2 ((Ben-David et al., 2010)). *For a fixed hypothesis h, if a random labeled sample of size m is generated by drawing βm points from $\mathcal{D}_S$ and $(1-\beta)m$ points from $\mathcal{D}_T$, and labeling them according to y_S and y_T respectively, then for any $\delta \in (0,1)$, with probability at least $1-\delta$ (over the choice of the samples),*

$$|\hat{\mathcal{L}}_\alpha(h,y) - \mathcal{L}_\alpha(h,y)| \leq \epsilon(m,\alpha,\beta,\delta), \tag{9}$$

where $\epsilon(m,\alpha,\beta,\delta) = \sqrt{\frac{1}{2m}\left(\frac{\alpha^2}{\beta} + \frac{(1-\alpha)^2}{1-\beta}\right)\log(\frac{2}{\delta})}$.

The detail function form of ϵ will be omitted for simplicity. We can fix m, α, β, and δ when the learning task is specified, then we can treat $\epsilon(\cdot)$ as a constant.

Theorem 1: Generalization Gap

Theorem 1. *Let $\mathcal{H}$ be a hypothesis class, and $\mathcal{S}$ be a labeled sample of size m generated by drawing βm samples from $\mathcal{D}_S$ and $(1-\beta)m$ samples from $\mathcal{D}_T$ and labeling them according to the true label y. Suppose $\mathcal{L}$ is symmetric and obeys the triangle inequality. Let $\hat{h} \in \mathcal{H}$ be the empirical minimizer of $\hat{h} = \arg\min_h \hat{\mathcal{L}}_\alpha(h, y)$ on $\mathcal{S}$ for a fixed $\alpha \in [0, 1]$, and $h_T^* = \arg\min_h \mathcal{L}_T(h, y)$ the target error minimizer. Then, for any $\delta \in (0, 1)$, with probability at least $1 - \delta$ (over the choice of the samples), we have*

$$\frac{1}{2}|\mathcal{L}_T(\hat{h}, y) - \mathcal{L}_T(h_T^*, y)| \leq \alpha\left(\mathrm{disc}_{\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T) + \lambda\right) + \epsilon, \tag{10}$$

where $\epsilon(m, \alpha, \beta, \delta) = \sqrt{\frac{1}{2m}\left(\frac{\alpha^2}{\beta} + \frac{(1-\alpha)^2}{1-\beta}\right)\log(\frac{2}{\delta})}$, *and* $\lambda = \min_{h\in\mathcal{H}} \mathcal{L}_S(h, y) + \mathcal{L}_T(h, y)$.

Proof of Theorem 1

Proof. We use Lemmas 1 and 2 for the bound derivation with their associated assumptions.

$$\mathcal{L}_T(\hat{h}, y)$$

$$\leq \mathcal{L}_\alpha(\hat{h}, y) + \alpha \left(\mathrm{disc}_\mathcal{H}(\mathcal{D}_S, \mathcal{D}_T) + \lambda\right), \quad (11)$$

(By Lemma 1)

$$\leq \hat{\mathcal{L}}_\alpha(\hat{h}, y) + \alpha \left(\mathrm{disc}_\mathcal{H}(\mathcal{D}_S, \mathcal{D}_T) + \lambda\right) + \epsilon, \quad (12)$$

(By Lemma 2)

$$\leq \hat{\mathcal{L}}_\alpha(h_T^*, y) + \alpha \left(\mathrm{disc}_\mathcal{H}(\mathcal{D}_S, \mathcal{D}_T) + \lambda\right) + \epsilon, \quad (13)$$

$$(\hat{h} = \arg\min_{h \in \mathcal{H}} \hat{\mathcal{L}}_\alpha(h))$$

$$\leq \mathcal{L}_\alpha(h_T^*, y) + \alpha \left(\mathrm{disc}_\mathcal{H}(\mathcal{D}_S, \mathcal{D}_T) + \lambda\right) + 2\epsilon, \quad (14)$$

(By Lemma 2)

$$\leq \mathcal{L}_T(h_T^*, y) + 2\alpha \left(\mathrm{disc}_\mathcal{H}(\mathcal{D}_S, \mathcal{D}_T) + \lambda\right) + 2\epsilon, \quad (15)$$

(By Lemma 1)

Cost of "swapping" is at least …

$$|\mathcal{L}_\alpha(h, y) - \mathcal{L}_T(h, y)| \leq \alpha \left(\mathrm{disc}_\mathcal{H}(\mathcal{D}_S, \mathcal{D}_T) + \lambda\right)$$

Specific sampling:

$$|\hat{\mathcal{L}}_\alpha(h, y) - \mathcal{L}_\alpha(h, y)| \leq \epsilon(m, \alpha, \beta, \delta)$$

Because $\hat{\mathcal{L}}_\alpha(\hat{h}, y) \leq \hat{\mathcal{L}}_\alpha(h_T^*, y)$

Specific "de-sampling":

$$|\hat{\mathcal{L}}_\alpha(h, y) - \mathcal{L}_\alpha(h, y)| \leq \epsilon(m, \alpha, \beta, \delta)$$

$$|\mathcal{L}_\alpha(h, y) - \mathcal{L}_T(h, y)| \leq \alpha \left(\mathrm{disc}_\mathcal{H}(\mathcal{D}_S, \mathcal{D}_T) + \lambda\right)$$