Smart Inference for Multidigit Convolutional Neural Network based Barcode Decoding

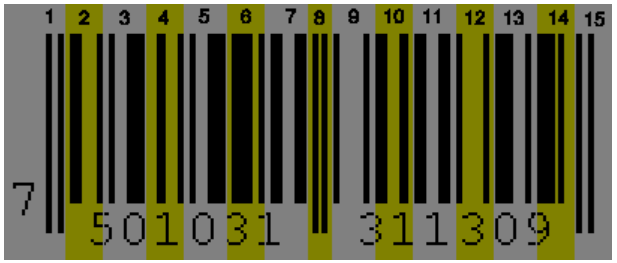
Thao Do, Yalew Tolcha, Taejoon Jun, Daeyoung, Kim - KAIST

Outline

- Barcode Decode & Its Problems
- Background Information
- Proposed Method
- Experiments
- Conclusion & Limitation

Barcode Decode & Its Problems

- Target ? (Focus on 1D barcode)
  - Decode cropped barcode images

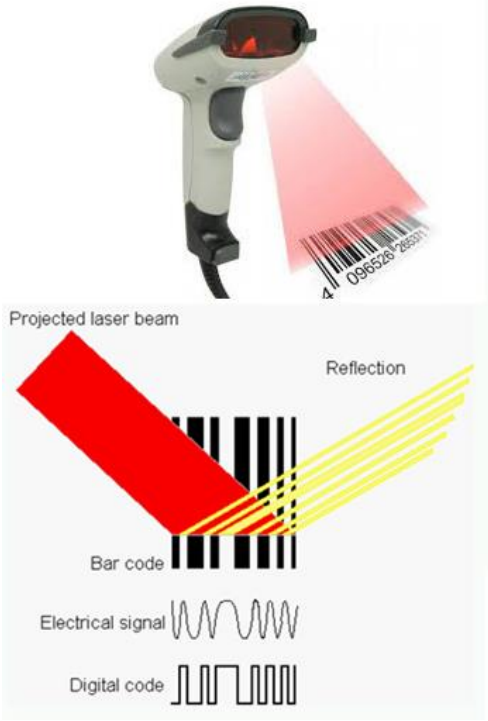


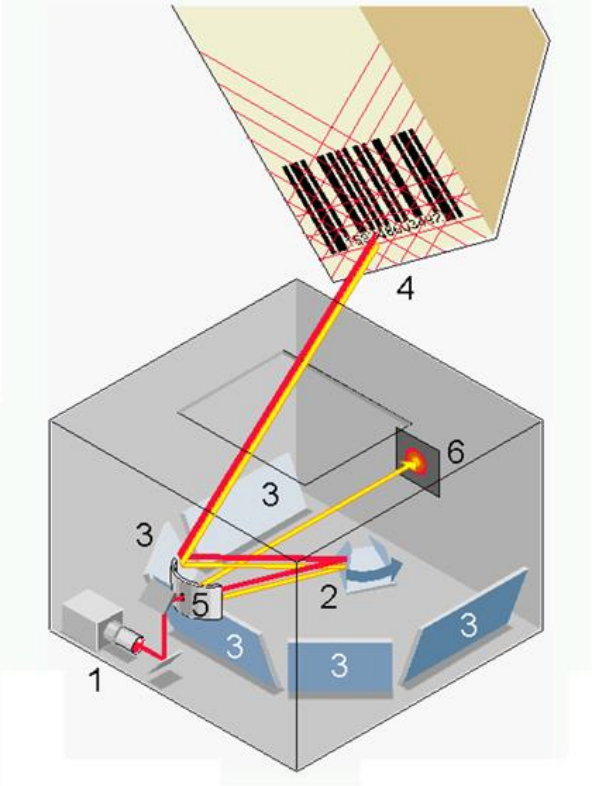
→ 7 501 031 311 309

- Traditional laser-beam approaches
- Conventional camera-based methods

Traditional laser-beam approaches

- Overheating
- Consume more energy
- Very short range





Source: *https://encyclopedia2.thefreedictionary.com/barcode+scanner*

Conventional camera-based methods

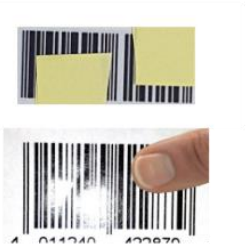
Apply various **static** filter functions to turn image to *good* bitmap image

→ **sensitive** to **cripple barcode** (*dark*, *occlusion*, *wrinkle*, *non-flat plane,* over light, blurry and …).

→ Why not try CNN? Actually there're work did it but with some limitations lead to low performance!

(a) customized shapes

(b) non-uniform covered



(c) double-obscured

(d) vertical rotated with scratched

(e) overexposed

(f) blurry with noise

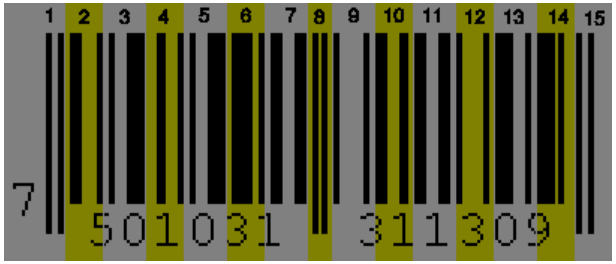
(g) dark/underexposed

(h) distort/wrinkled

Background Information

- Each barcode family has own composition and decoding rule.
- But most of them has *checksum* to self-validate

$$(D[L-0]*\mathbf{1}+D[L-1]*\mathbf{3}+D[L-2]*\mathbf{1}+D[L-3]*\mathbf{3}+\ldots$$
$$D[L-2i]*\mathbf{1}+D[L-2i-1]*\mathbf{3}+\ldots D[1]*\mathbf{1}\text{ if }L \bmod 2 == 0\text{ else }*\mathbf{3}) \bmod 10 = 0$$

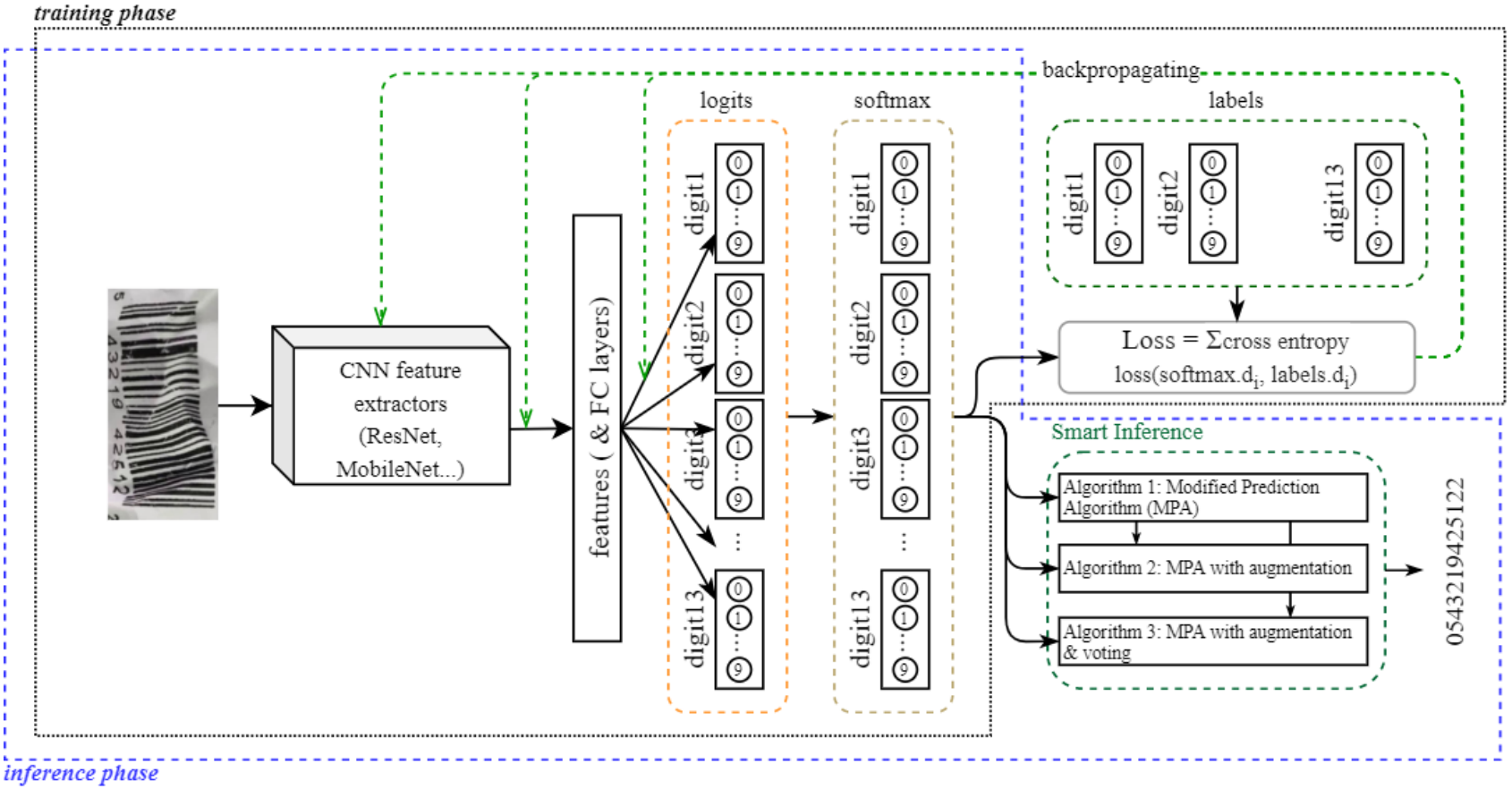


(digit1***1*** +digit2***3*** + digit3***1*** + digit4***3*** + digit5***1*** + digit6***3*** + digit7***1*** +digit8***3*** + digit9***1*** + digit10***3*** + digit11***1*** + digit12***3*** + digit13***1***) **% 10** = **0**

(7***1*** + 5***3*** + 0***1*** + 1***3*** + 0***1*** + 3***3*** + 1***1*** + 3***3*** + 1***1*** + 1***3*** + 3***1*** + 0***3*** + 9***1***) % 10 = 0

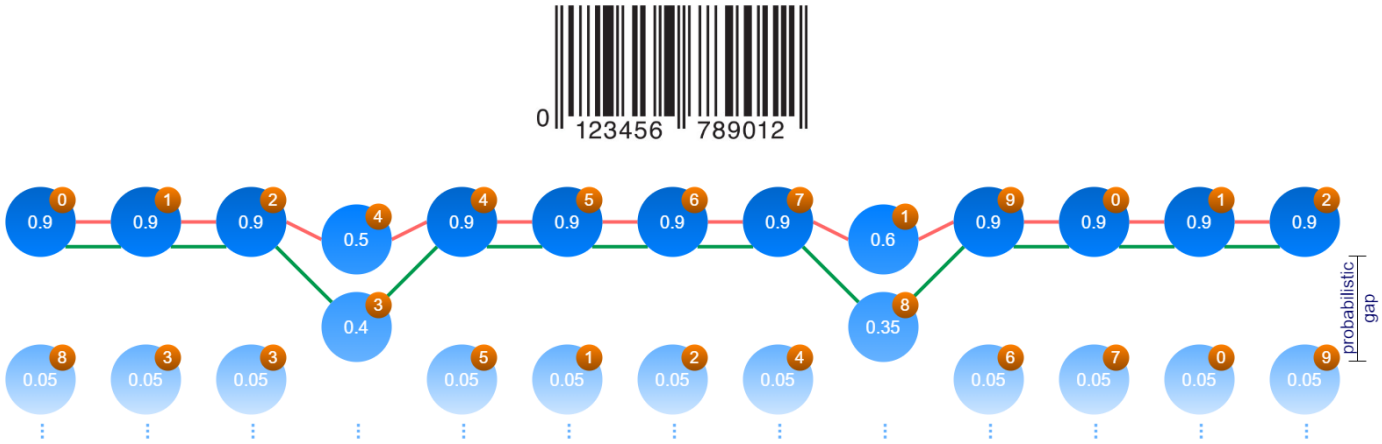
Proposed Method – Overview

Based method proposed by Fred et al[1] inspired by Goodfellow et al [2]



Inspiration

"The predicted value with the **highest probability** is *not always* the **correct value**"



Smart Inference (Algo 1)

1/ Calculate the differences (**gap**) between the 2 highest probabilities of each digit.

2/ Sort ascending the gaps, choose **MaxIter** *smallest* gaps.

(bigger gap ~ more confidence of the model on the outcome of highest probability outcome, less confidence of its for outcome of the 2nd highest,
smaller gap ~ higher chance the outcome with 2nd highest probability is actually the correct of the digit at a position)

3/ Create all combinations from those gaps, sort the list of all combinations in the **descending** order of sum of probabilities.

4/ Test one by one and stop at the *checksum equation-satisfied* combination, this is the predicted sequence; if no combination satisfy, return no barcode!

Algorithm 1: Modified Prediction Algorithm (MPA)

Input : Trained model (Net), Image (Img), MaxInter
Output : Barcode
Compute logit[][] using Net given Img;
for $k \leftarrow 1$ **to** L **do**
  $Prob^k[] \leftarrow softmax(logit^k[])$;
  sort $Prob^k[]$ **desc**;
  append $Prob^k[0].val - Prob^k[1].val$ to Gap;
  append k to Gap_pos;
  $digit^k \leftarrow Prob^k[0].index$;
end
sort Gap_pos **asc** with Gap;
$init\_sequence \leftarrow$ join($digit[]$);
append $init\_sequence$ to seq_list;
for $i \leftarrow 0$ **to** $(MaxIter - 1)$ **do**
  $k \leftarrow Gap\_pos[i]$;
  $second \leftarrow Prob^k[1].index$;
  for *each* $seq \in seq\_list$ **do**
    $new\_seq \leftarrow$ modify seq at position k with $second$;
    append new_seq to new_seq_list;
  end
  append new_seq_list elements to seq_list;
end
sort seq_list **desc** with sum_probability(seq_list);
for *each* $seq \in seq\_list$ **do**
  if *checksum(seq)* **then**
    return seq;
  end
end
return *null*

Smart Inference + test-time augmentation (Algo 2)

Idea?

- Human also turn their heads a bit to recognize things sometimes

Why [0°, 90°, 180°, 270°]?

- Simple, less calculation,
- No losing more information

Why stop at first satisfied sequence?

- Fast
- Empirical experiments show it's also effective

Algorithm 2: MPA with Augment

Input : Trained model (Net), Image (Img), MaxInter

Output: Barcode

append rotate(Img, [0°, 90°, 180°, 270°]) to *img_list*;

for *each img* ∈ *img_list* **do**

  seq ← Predict by **Algo_1**(Net, *img*);

  if *seq not null* **then**

    return *seq*;

  end

end

return *null*

Smart Inference + test-time augmentation + voting (Algo 3)

Idea?

- Pick the sequence which is more consistently predicted by variant inputs
  (No matter which rotation applied, barcode must be the same!)

Algorithm 3: MPA with Augment and Vote

Input : Trained model (Net), Image (Img), MaxInter

Output: Barcode

append rotate(Img, [0°, 90°, 180°, 270°]) to *img_list*;

for *each img* ∈ *img_list* **do**

  seq ← Predict by **Algo_1**(Net, *img*);

  append *seq* to *seq_list*

end

if *seq_list not empty* **then**

  frequency(*seq_list*);

  voted_seq ← *seq* with highest count;

  return *voted_seq*;

else

  return *null*;

end

Experiments
- Dataset (Real captured)

Real dataset with variety of hard conditions:

- 2500 barcode images
- 2000 validation
- 500 for train → augmented to 20,000 samples

Comprised of:

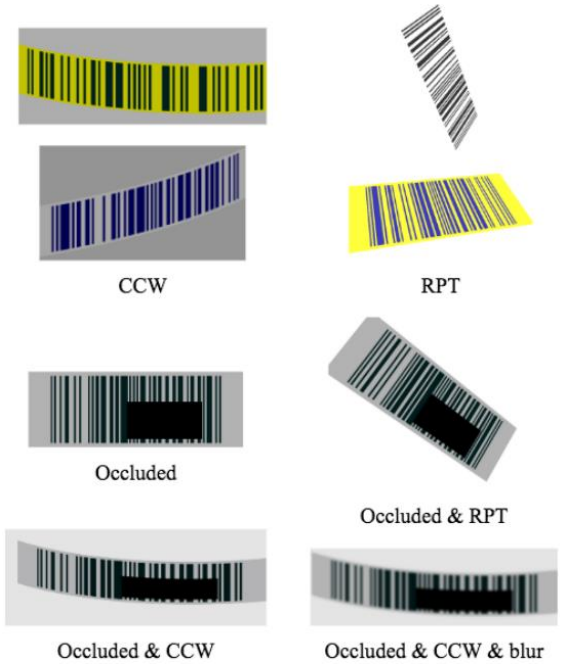
- 1055 from MuensterDB [3]
- 408 Zamberletti et al [4]
- 1037 our collection
- From 5 supermarket (3 countries)
- Various type: food, books, kitchenware, clothes, stationary.
- Various material: metal cans, wine bottles, food plastic bags, cardboard box.
- Various light: florescent, incandescent bulb, morning and afternoon.
- Auto-focus off, hand shaking, long distance, obscured by fingers, wrinkled, distort.
- We double checked, draw box, labelled.

Experiments
- Dataset (Synthesized)

Synthetics set:

- 250,000 samples for various conditions

Total: 270,000 for training.



Condition(s)	*# of samples*
norm	30000
dark	30000
occluded	20000
occluded+dark	20000
Rotated, Perspective Transformed	20000
RPT+dark	20000
Cylindered, Curvy Warped	20000
CCW+dark	20000
occluded+RPT	5000
blur	5000
RPT+blur	5000
CCW+blur	5000
upside down	6000
upside down+dark	6000
upside down+blur	6000
upside down+CCW	6000
upside down+occluded	6000
heavy noise+rotated	2000
exposed+occluded+RPT+CCW	6000
dark+occluded+RPT+CCW	6000
occluded+RPT+CCW	6000

Experimental setups

- Input size: 285x285, batch size: 32
- The training processes were made using NVIDIA Titan RTX with 24 GB VRAM.
- Evaluation experiments were conducted on desktop using Intel Core i9 9900KF processor, 32GB RAM
- Low-computational experiments were run on a NVIDIA Jetson Nano board with CUDA-enabled using NVIDIA TensorRT models (converted from PyTorch).

Experimental results

TABLE II
TOOL & MODELS WITHOUT MPA PERFORMANCES

Model	*Accuracy*	*CPU (ms)*	*# of params (M)*
Zxing	58.25%	**7.65**	NA
Dynamsoft	93.10%	978.8	NA
Google API	82.45%	211.9	NA
Cognex	84.60%	111.9	NA
ResNet50	**93.35%**	66.5	99.5
MobiletNetV2	72.25%	32.4	**15.7**
MobiletNetV2_kd	83.45%	32.4	**15.7**
DenseNet169	84.90%	75.65	30
ResNet34	88.70%	38.3	40.7
ResNet34_kd	89.20%	38.3	40.7
Fridborn-similar	31.85%	104.9	403.3
Non-residual	80.80%	103.2	78.5

TABLE III
USING MPA PERFORMANCES

	Model	*nonMPA*	*max=1*	*max=2*	*max=3*	*max=4*
Accuracy	ResNet50	0.9335	0.942	0.9435	**0.9445**	0.9445
	MobiletNetV2	0.8345	0.8595	**0.869**	0.868	0.8645
	DenseNet169	0.849	0.8645	0.876	**0.8775**	0.874
	ResNet34	0.892	0.9075	0.911	**0.912**	0.911
	Non-residual	0.808	0.8295	0.844	**0.8455**	0.841
# of errors	ResNet50	133	**31**	48	77	106
	MobiletNetV2	331	**59**	106	164	241
	DenseNet169	302	**54**	105	176	230
	ResNet34	216	**41**	73	116	157
	Non-residual	384	**55**	104	197	285

TABLE IV
USING MPA & AUGMENTATION PERFORMANCES

	Model	*nonMPA*	*max=1*	*max=2*	*max=3*	*max=4*
Accuracy	ResNet50	0.9335	**0.958**	0.956	0.951	0.946
	MobiletNetV2	0.8345	**0.906**	0.8975	0.8855	0.8705
	DenseNet169	0.849	**0.9155**	0.9075	0.8915	0.877
	ResNet34	0.892	**0.9375**	0.9315	0.9235	0.916
	Non-residual	0.808	**0.89**	0.879	0.861	0.8445
# of errors	ResNet50	133	**56**	73	95	108
	MobiletNetV2	331	**121**	182	225	259
	DenseNet169	302	**113**	172	216	246
	ResNet34	216	**95**	129	152	168
	Non-residual	384	**145**	212	274	311

TABLE V
USING MPA & AUGMENTATION WITH VOTING PERFORMANCES

	Model	*nonMPA*	*max=1*	*max=2*	*max=3*	*max=4*
Accuracy	ResNet50	0.9335	**0.9585**	0.9585	0.9525	0.95
	MobiletNetV2	0.8345	**0.9085**	0.906	0.899	0.8945
	DenseNet169	0.849	**0.9125**	0.8985	0.8785	0.8545
	ResNet34	0.892	**0.933**	0.93	0.9215	0.911
	Non-residual	0.808	**0.8855**	0.8735	0.853	0.8355
# of errors	ResNet50	133	**55**	68	92	100
	MobiletNetV2	331	**116**	165	198	211
	DenseNet169	302	**119**	190	242	291
	ResNet34	216	**104**	132	156	178
	Non-residual	384	**154**	223	290	329

Notes

- Real-collected dataset cause **too big loss** for training from scratch → using transfer learning after the model converged a bit on synthesized data.

Conclusion & Limitation

- Contribution
  - *Smart Inference* in prediction phase really boosted the accuracy of the multi-digit CNN models.
  - Open EAN13-decoding dataset of 2500 real-collected cropped images under various wild practical conditions.
  - Minimized the model to work well on a small board (Jetson Nano) to demonstrate its applicability in practical use using Knowledge Distillation.
- Limitation
  - Still need the component to localize barcode regions
  - Only work with fixed maximum length barcode families
    → Too long Code39 type can't be used.

References

- [1] Reading barcodes with neural networks, 2017, ISRN: LiTH-ISY-EX--17/5102--SE
- [2] Ian J Goodfellow, Yaroslav Bulatov, Julian Ibarz, Sacha Arnoud, and Vinay Shet. Multi-digit number recognition from street view imagery using deep convolutional neural networks. arXiv, preprint arXiv:1312.6082, 2013.
- [3] Steffen Wachenfeld, Sebastian Terlunen, and Xiaoyi Jiang. Robust recognition of 1-d barcodes using camera phones. In 2008 19th International Conference on Pattern Recognition, pages 1–4. IEEE, 2008.
- [4] Alessandro Zamberletti, Ignazio Gallo, Moreno Carullo, and Elisabetta Binaghi. Decoding 1-d barcode from degraded images using a neural network. In International Conference on Computer Vision, Imaging and Computer Graphics, pages 45–55. Springer, 2010.

Thank for listening

And stay safe