ELEVEN SUPERCASE DATA CHALLENGE

# OPTIMIZER

# IMAGE RECOGNITION

GROUP 8

# Presentation agenda

01 Our Team

02 Problem Definition

03 Solution 1 – Digit Cropping

04 Solution 2 – End-to-End solution

05 Conclusion and Next Steps

# The team that has been working on the challenge

**Fernando Pérez**

**Jørgen Lund**

**Jiahao Wang**

**Ching-Yu Lin**

**Akshay Sundar**

# Our challenge is automatically extracting digits from images

**01** Pictures taken before and after fuel transaction

**02** AI Model to detect numbers

**03** Compute transaction volume

# How are we measuring success?

## Digit-by-digit Accuracy

Common metric to compute the accuracy of model.

$$\frac{sum\ of\ digit\ wise\ success}{sum\ of\ total\ number\ of\ digits}$$

## Mean Absolute Error

Proposed business metric to compute distance from true value

$$\sum \left| \frac{prediction\ - label}{\#\ of\ pictures} \right|$$

# We have developed an AI solution that can extract volumes from screen pictures

Digit extraction solution

End-to-end solution

Analog pictures

Digital pictures

Generated digits

Digit cropping

Model

8

Prediction digit-by-digit

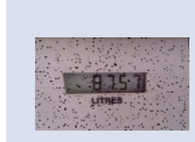
00189

Pre-trained Model + fine-tuning

Prediction

19400

Augmented images



Generated images

# Screen needs to be cropped before performing the digit detection correctly

**Original picture** → **Processed picture** → **Cropped picture**

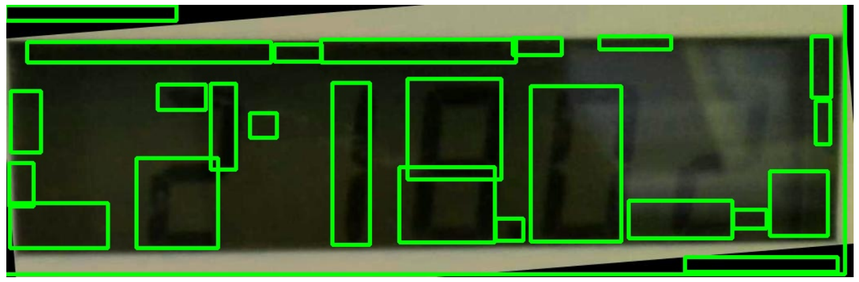


- Gray scale
- Gaussian Blur
- Dilation
- Inversion

- Rotation
- Re-crop

# Two approaches to detect digits…

## Finding contours



### Illumination

Exposure
Contrasting

### Dimension reduction

Gray Scaling
Adaptive
Thresholding

### Noise reduction

Bilateral
Filtering
Gaussian
Blurring

### Smoothing

Eroding
Dilating
Closing

## Enforcing contours

### Heuristic methods

Width-height ratio
distribution of
cropped images

# We also built a pipeline to detect digits in the analog images

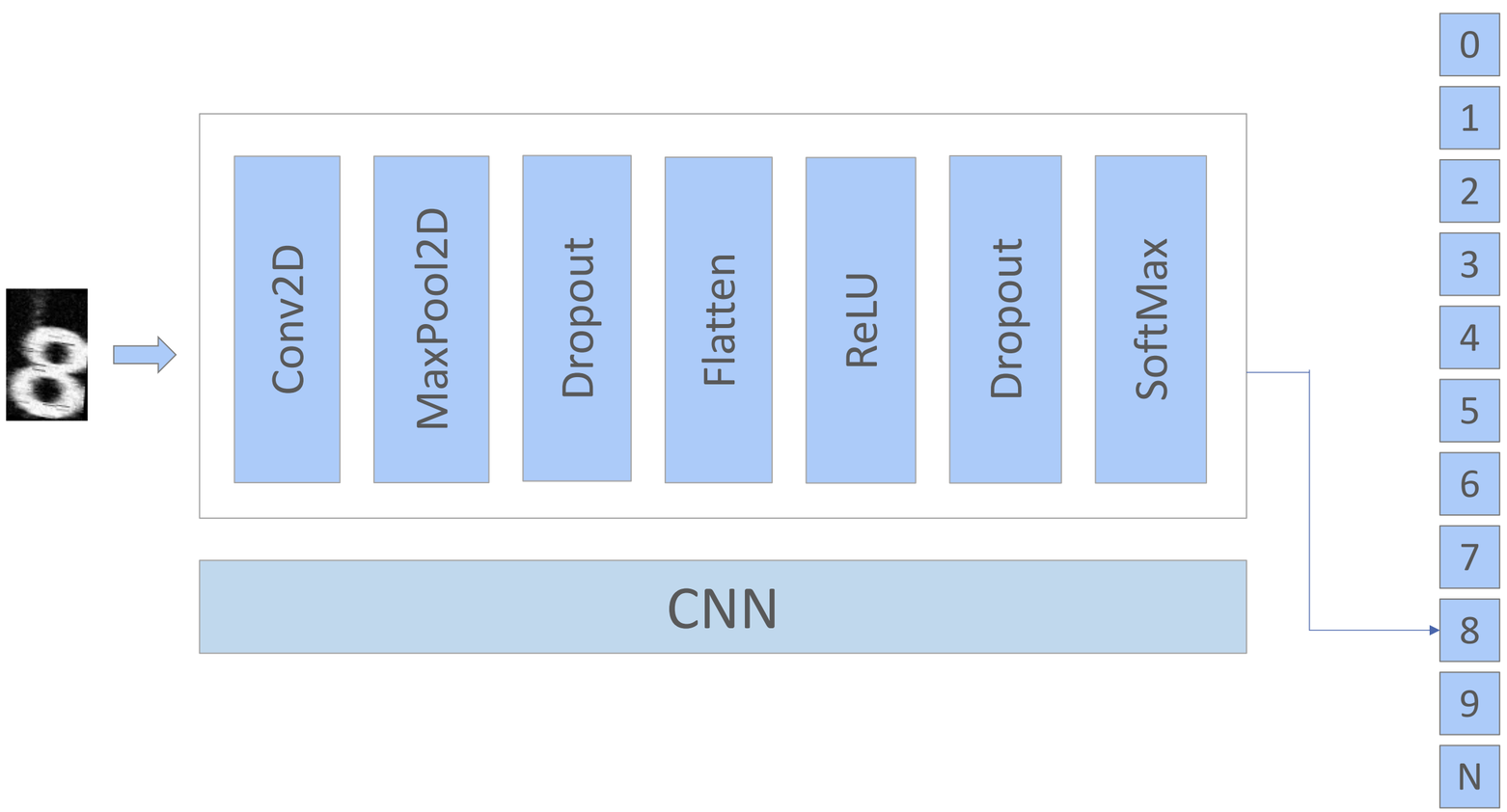

Gray scale
Gaussian Blur
Adaptive Thresholding
Erosion


Ratio Filtering
Area Clustering


Ratio Filtering
Area Filtering
Height Filtering
Y-axis Filtering


Y-axis Clustering
Area Clustering
Box Enforcement

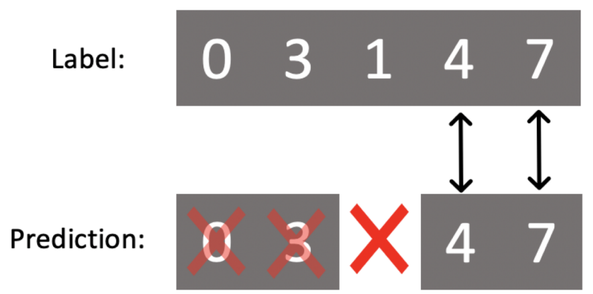
# A CNN model was adapted for the analog digit recognition problem, ultimately yielding a digit by digit accuracy of 59.45%



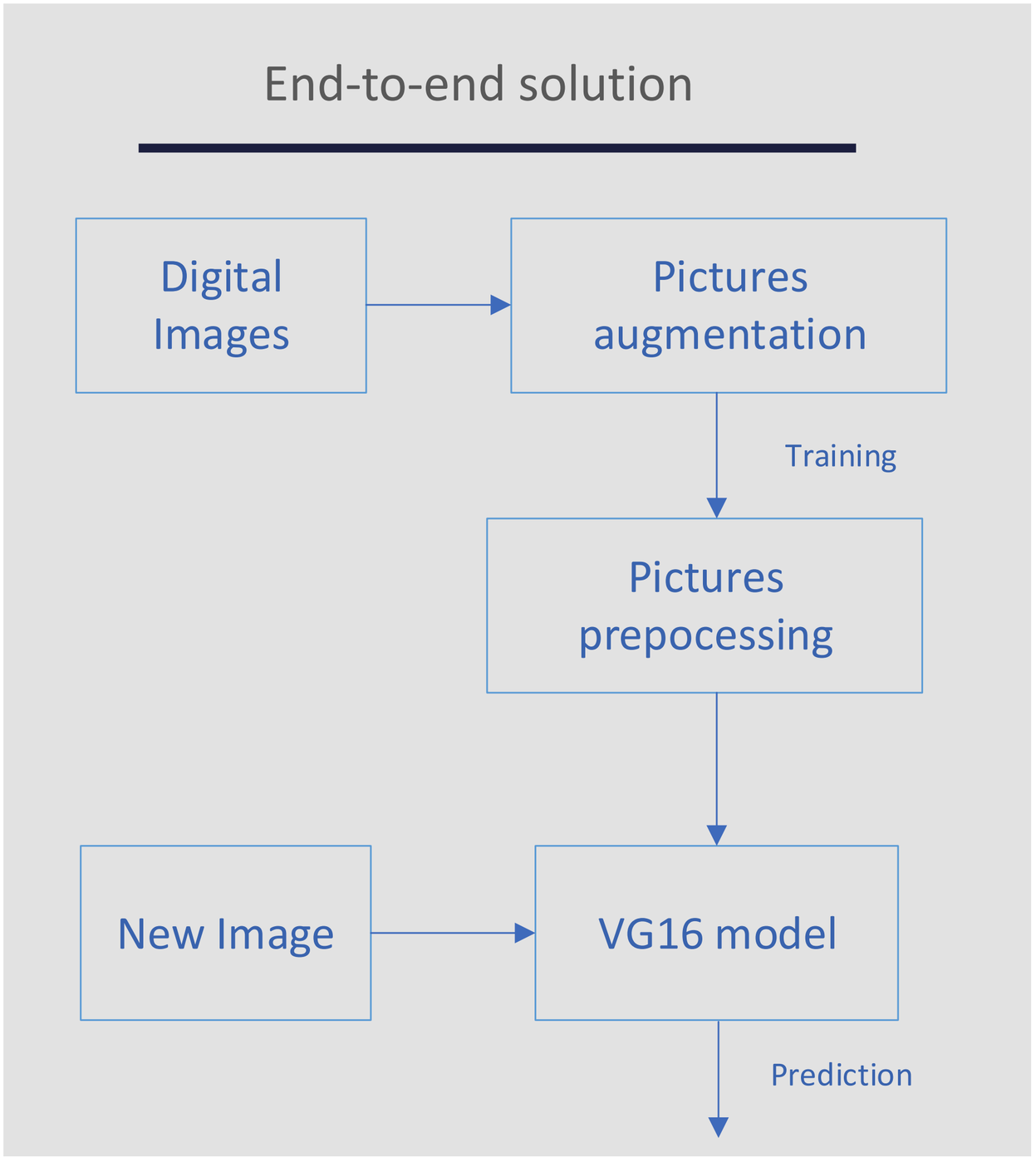

## Highlights

- The model additionally includes a class for “No Digit”
- The trained CNN model has an accuracy of **68.4%** on the test set

Based on a digit by digit comparison of the predicted number with the test labels, the accuracy of the overall test set is calculated at **59.45%**



# Due to the complicated extraction of the digits, we have worked on an alternative end-to-end solution



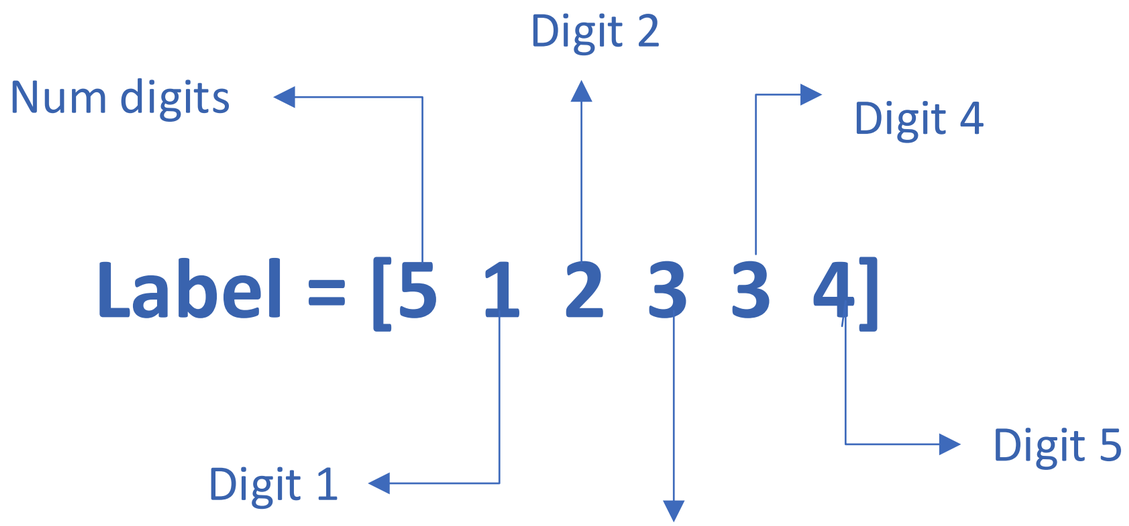

- It has many advantages.
  - Clean structure
  - Easy to extend
  - High accuracy
- But it has some challenges
  - Lack of annotated data
  - Easy to overfit
  - Computation power

# The training data is not labelled precisely, as decimal numbers are not included and some are wrong




| Original label | New label |
|---|---|
| 194 | 194.00 |
| 37 | 37.90 |
| 59 | 59.0 |

# The labels need to be pre-processed before going through the model




Number of digits: between 0 and 5. The maximum number of digits is 5.Therefore, the maximum amount of litters detected is 999.99

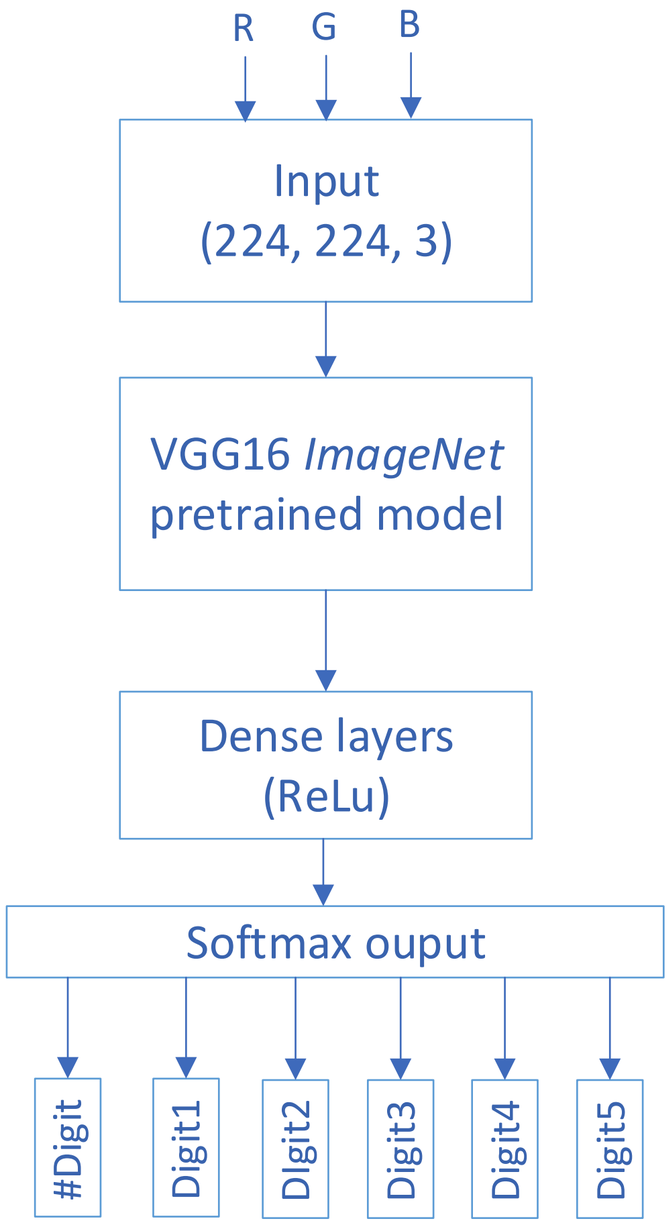
If a number is not present in the screen, we replace it by 10.

61.31 → [ 4 10 6 1 3 1 ]

61.3 → [ 3 10 6 1 3 10 ]

100 → [ 3 1 0 0 10 10 ]

# Due to the small amount of real images, we used ImageNet pretrained model




- Each image has been resize and saved as 224x224 pixel RGB.
- The pre-trained VGG16 used the ‘ImageNet’ weights which were trained over 1000 classes.
- Dense layers have been added. Activation function used was ReLu for all the layer except the final dense layer.
- Six parallels element output are connected to the final layer. The first element represent the  number of digits, followed by the five digits.

# First approach: run raw images through an augmentation sequence

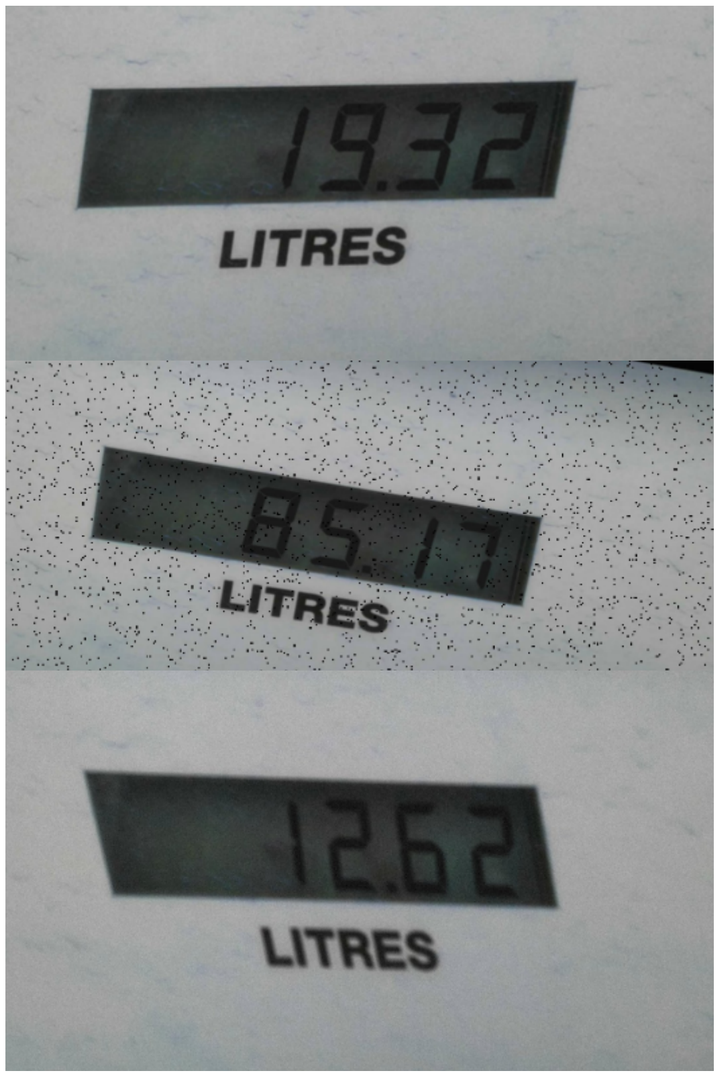
**Raw Images** → **Augmentation** → **Set of New Images**


- Noise
- Rotation
- Shear
- Blur
- Dropout

# Second approach: create synthetic data using templates

**Raw Image** → **Template** → **Set of New Images**





# In training, we tried different combinations of original and synthetic data

Raw Data

Raw Data | Augmented Data

Raw Data | Augmented Data | Synthetic Data

~800 samples

~10k samples

~10 k samples

# The model with better accuracy is trained using the raw + augmented data

| | Raw + Augmented + Synthetic | Raw + Augmented |
|---|---|---|
| | **62.35%** of accuracy digit by digit | **64.99%** of accuracy digit by digit |
| | Mean absolute error: **41.08 litters** | Mean absolute error: **37.63 litters** |
|  | [ 4 10 5 4 0 0 ] | [ 4 10 5 4 0 0 ] |
|  | [ 5 1 2 4 0 0 ] | [ 5 1 2 5 0 0 ] |

# This model has higher accuracy for all the different quality of pictures: HQ, MQ and LQ

| | Raw + Augmented + Synthetic | Raw + Augmented |
|---|---|---|
| HQ | **54.1% of accuracy** digit by digit<br>Mean square error: **49.29 liters** | Digit by digit: **55% of accuracy**<br>Mean square error: **45.73 liters** |
| MQ | **66.4% of accuracy** digit by digit<br>Mean square error**: 38.82 liters** | Digit by digit: **66.4% of accuracy**<br>Mean square error: **29.6 liters** |
| LQ | **65% of accuracy** digit by digit<br>Mean square error: **37.18 liters** | Digit by digit: **68.15% of accuracy**<br>Mean square error: **37.77 litters** |

# We explored various approaches and have different ideas to further implement

## Highlights

**01** Two solutions

Cropping method and end-to-end method

**02** Data

Creation of methods to augment the training data.

**03** Cropping

Cropping method works for high quality data

**04** End-to-end

End-to-end has the potential to generalize better and works better for low quality data

## Next steps

- Train end-to-end model with more (synthetic) data and with more computation power
- Generate the numbers with similar distribution
- Workshop to train workers to improve data capture
- Aggregate the data at regional/ national level to make business decision.

# Appendix: Unbalanced test set

Balanced validation set

**CNN class-wise accuracy**

| | | |
|---|---|---|
| 0 | → | 90% |
| 1 | → | 80% |
| 2 | → | 70% |
| 3 | → | 70% |
| 4 | → | 80% |
| 5 | → | 60% |
| 6 | → | 70% |
| 7 | → | 90% |
| 8 | → | 30% |
| 9 | → | 60% |
| Non-digit | → | 82% |

Unbalanced test set

**CNN class-wise accuracy**

| | | |
|---|---|---|
| 0 | → | 82.3% |
| 1 | → | 80% |
| 2 | → | 60% |
| 3 | → | 60% |
| 4 | → | 100% |
| 5 | → | 100% |
| 6 | → | 100% |
| 7 | → | 0% |
| 8 | → | 16.7% |
| 9 | → | 0% |
| Non-digit | → | 55.6% |