2020 Intelligent Sensing Summer School, September 1-4

The CORSMAL challenge

Team NTNU - ERC

Title of the solution

Guilherme Christmann
Jyun-Ting Song
You-Sheng Liao
Greene Chang

CORSMAL
Collaborative object recognition, shared manipulation and learning

Task #1 - Filling type classification

- Classification is challenging through vision data.
  - Opaque containers.
  - Scenarios where the filling is poured outside of view.
  - No labels for the localization of containers.

- We chose to classify using just **audio data**.
  - Transparency of containers doesn't matter.
  - Sound data is (pretty much) the same even if poured out of view.
  - Already have all the necessary labels.

Task #1 (Filling type) - Feature Extraction

- We compute the **MFCC features** of the audio using *librosa**.
  - Number of MFCCs: 40
  - Window size: 20 ms (441 samples @ 22kHz)
  - Maximum length = 30 seconds
  - Normalize each MFCC by its **mean** and **std** over the sequence.

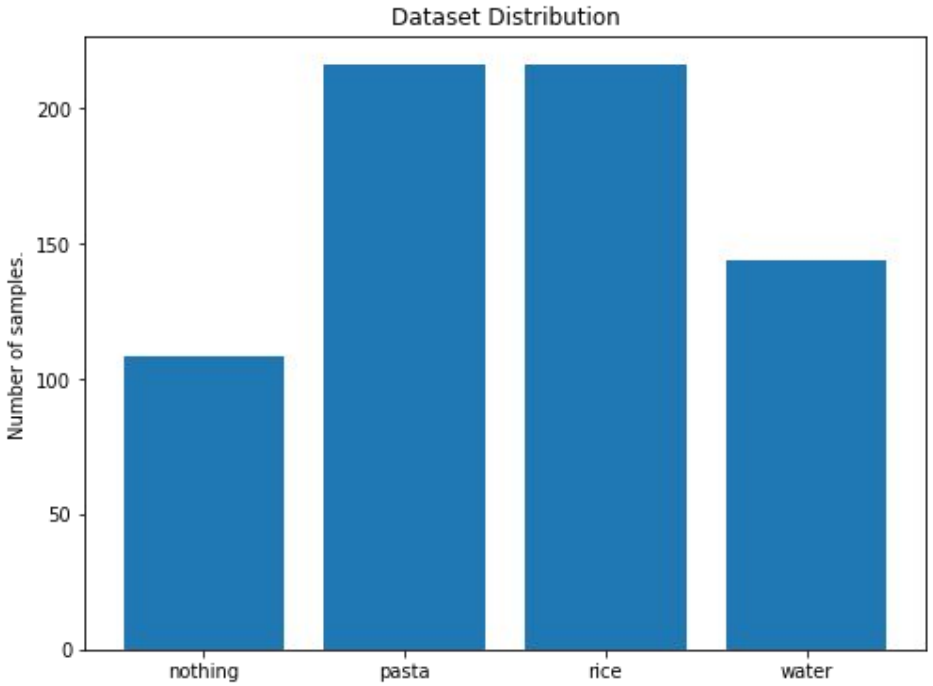
```
# Normalize the sequence according to its own data
### Normalization for each MFCC individually
_mean = np.mean(sequence, axis=0)
_std = np.std(sequence, axis=0)

return (sequence - _mean) / _std
```

  - Sequences are zero-padded to the largest sequence.

* https://github.com/librosa/librosa

Task #1 (Filling type) - Train-Val Split

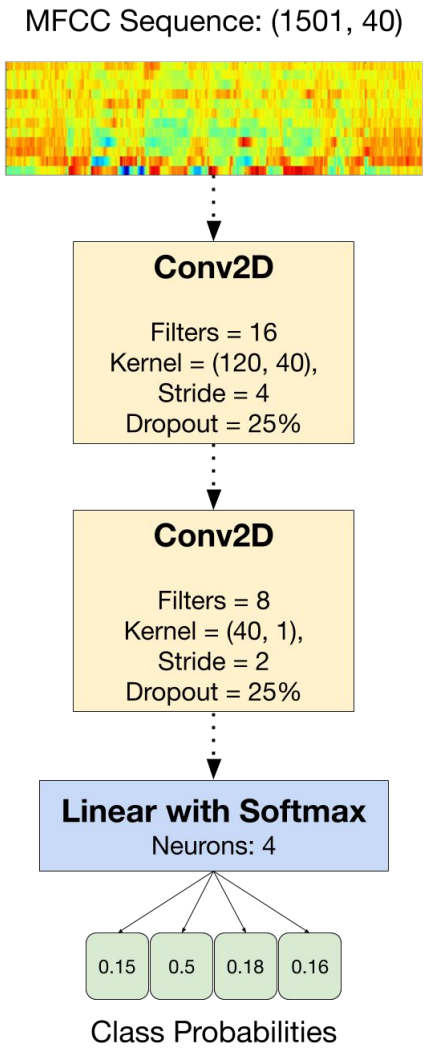
- Loading all audio samples provided in training set:
  - 684 audio files, padded to sequence length 1501, each with 40 MFCC. Tensor shape: [684, 1501, 40]
  - Split 10% of samples for validation.
  - 615 for training and 69 for val.



- Dataset is unbalanced.
  - Weigh each class according to its number of samples.
  - Less samples -> Larger weight

Task #1 (Filling type) - Model Definition and Training

- Simple conv based model.
  - Input is the MFCC sequence.
  - 2 conv layers followed by 1 linear layer.
  - Softmax classification.
  - **~86k parameters.**
- SGD optimizer (lr=0.00025, momentum=0.9)
- Cross Entropy Loss with class weights.
- Batch Size of 16.
- Trained for 200 epochs (<= 1 minute real-time).



Got a good result on val set, but can't be trusted.

Small sample size and no cross validation

	precision	recall	f1-score	support
nothing	1.00	1.00	1.00	11
pasta	1.00	1.00	1.00	25
rice	1.00	1.00	1.00	20
water	1.00	1.00	1.00	13
accuracy			1.00	69
macro avg	1.00	1.00	1.00	69
weighted avg	1.00	1.00	1.00	69

Test Acc: 100.000%

Task #2 - Filling level classification

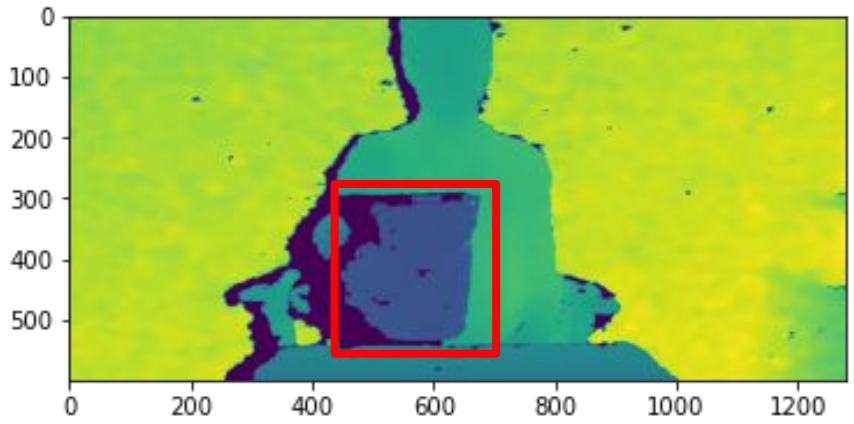
- We didn’t realize Task #2 due to time constraints.

- We believe it was the most challenging task.

Task #3 - Container volume estimation

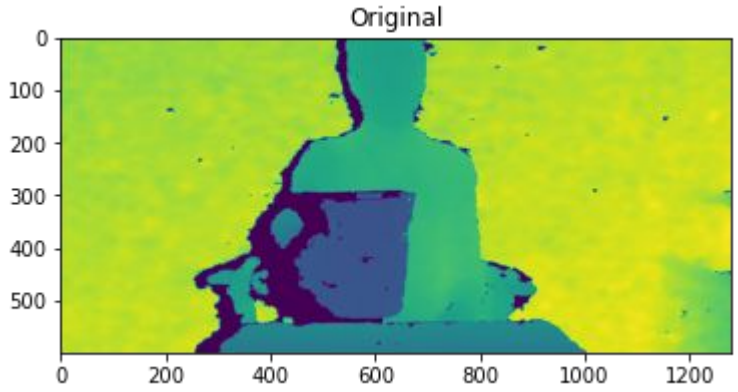
- Initially, we wanted to use a pre-trained SoTA object detector (e.g. MaskRCNN) to detect the container in RGB frame and retrieve the ROI from the equivalent multi-view Depth frames.
  - Due to time constraints we couldn't get it running.
- So, we took a simpler approach using **only the Depth data** from one camera (c3 - front facing person).

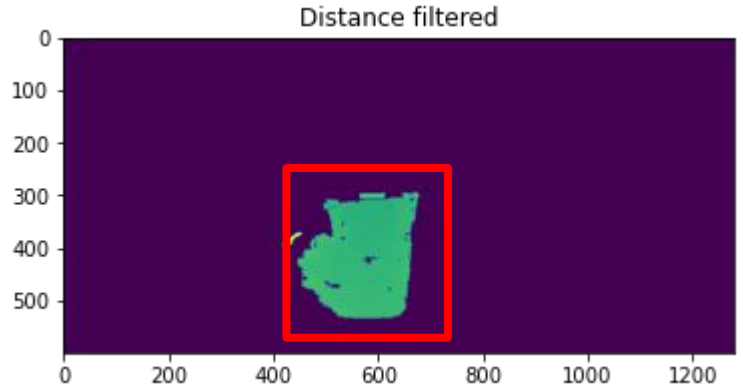
- Our method relies on **extracting the region where the object** is localized in the depth image.
  - We feed the extracted **ROI pixel (depth)** values along with the **size of the region** to an NN model.

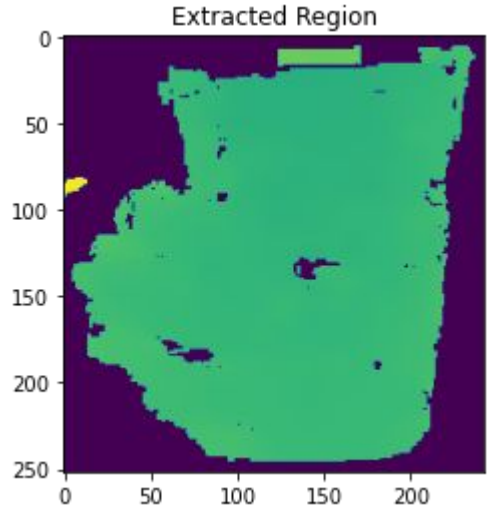


Task #3 (Volume estimation) - ROI Extraction

- In every video towards the end, the person will **extend the arm holding the container,** showing it to the camera.
- Based on this, we can “filter” the video by only considering values up to a distance (MAX_DIST=700 mm).
  - We start looking at the last frame, and work backwards until we find a reasonable ROI.
  - For each image, we find large contour and its bounding box (expanded a bit 5%).

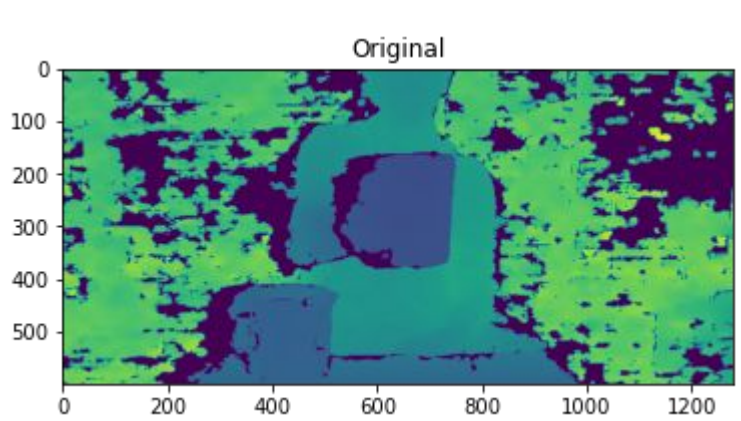


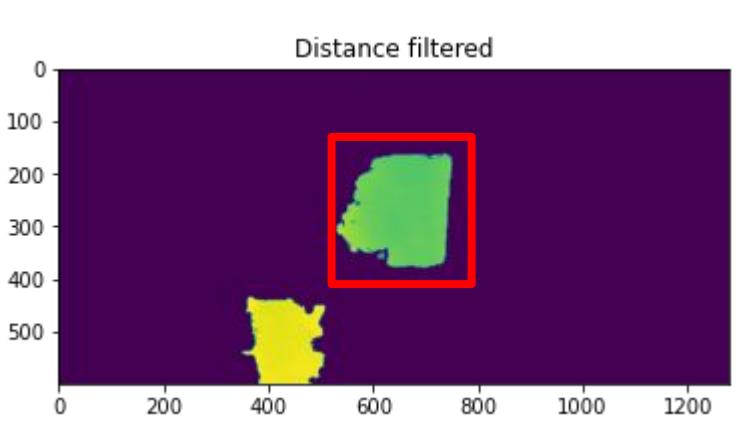


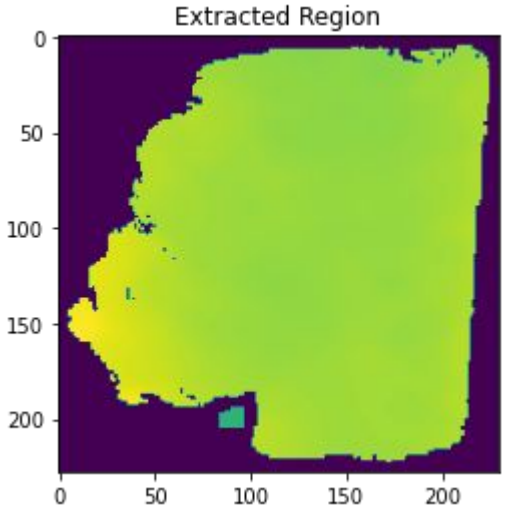


Task #3 (Volume estimation) - ROI Extraction

- Sometimes, there will be two large contours in the image, if the jug is also close enough.
- In this case, we retrieve the contour **closest** to the camera.







Task #3 (Volume estimation) - Dataset construction

- The dataset **inputs** are the **extracted ROIs** as well as the **ROI's dimensions**. The **targets** are the container volumes.
  - The ROIs are resized to (112x112) and normalized by the distance (divided by 700).
  - The ROI's dimensions are normalized by the full depth image size [0~1].

  - Targets are scaled dividing by 4000.

- Total of 672 samples:
  - Out of 684 videos, failed to find a reasonable ROI in 12.
  - 15% validation split. 100 samples for validation, and 572 for training.

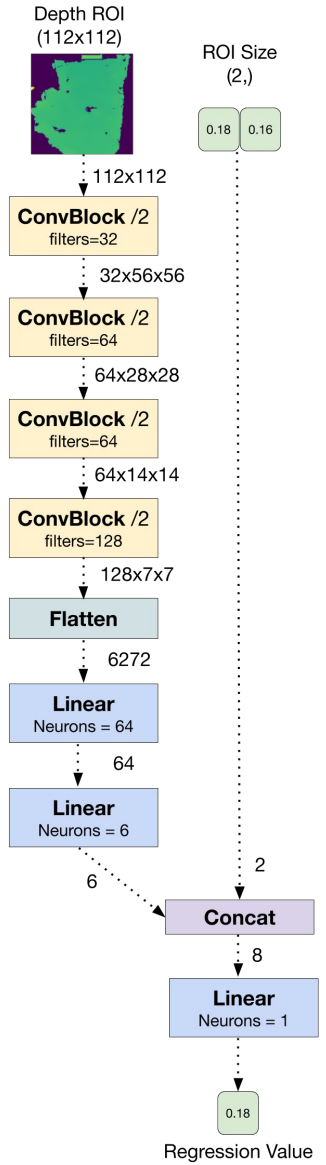
Task #3 (Volume Estimation) - Model Definition and Training

- 4-conv batchnorm layers + 3-linear layers model.
  - Two inputs: Depth-ROI and ROI's dimensions.
  - Max Pooling to reduce dimension
  - Batchnorm between every layer.
  - ReLU activation
  - **~532k parameters.**

- Adam optimizer (lr=0.00025)
- MSE Loss.
- Batch Size of 8.
- Trained for 200 epochs (<= 2 minutes real-time).

Results definitely not good.

Input data from only 1 depth view is probably not enough for accurate estimation of container volume.



Conclusion

- We believe we got a reasonable performance in Task 1, but because of time constraints couldn’t develop a good result for Task 2 and 3.

- Using multi-modal data is a **must** in order to achieve a good performance for tasks 2 and 3, but data pre-processing can be troublesome.

- We would like to experiment with SoTA detectors on these problems, as well as 3D reconstruction models for estimating the volume.

Hardware Specifications and Group

- Ran everything on a local computer in our lab.
  - GPU: 1x 1080 Ti
  - RAM: 32 GB.
  - CPU: 6 cores - i7 8700 3.20 GHz

- Software: librosa, PyTorch, OpenCV.

- Group distribution:
  - Guilherme Christmann (Master's student, general ideas and implementation of the models).
  - Jyun-Ting Song (Undergraduate student, general ideas and data loading and pre-processing)

  - Other members are also undergraduates, but couldn't contribute much due to other school responsibilities.

Thanks for your attention.