Università
degli Studi
di Padova

Crowd Counting

Convolutional Neural Networks of Density Maps

Alberto Schiabel, Linpeng Zhang

Vision and Cognitive Services • July 2020

Table of contents

Alberto Schiabel

Table of contents

Alberto Schiabel

Linpeng Zhang

What is crowd counting?

It's the task of estimating the number of people in an image.

Crowd counting is challenging due to:

- Scale variations
- Limited size of available datasets
- Non-uniform distribution of people
- Object occlusions
- Illumination issues
- Blurred frames

What is crowd counting?

It's the task of estimating the number of people in an image.

Crowd counting is challenging due to:

- Scale variations
- Limited size of available datasets
- Non-uniform distribution of people
- Object occlusions
- Illumination issues
- Blurred frames

Motivation

Crowd counting is useful for:

- Overcrowding detection
- Crowd scene analysis
- Video surveillance

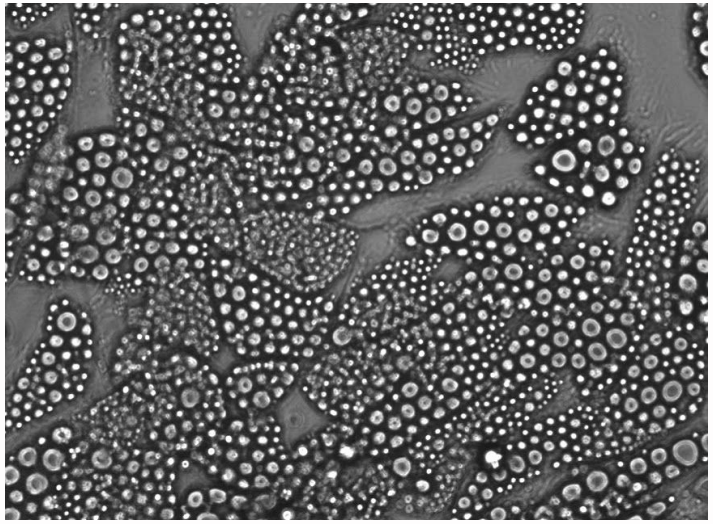
Motivation

Crowd counting is useful for:

- Overcrowding detection
- Crowd scene analysis
- Video surveillance

It can also be generalized to other problems:

- Vehicle counting
- Biological cell counting



Traditional approaches

Detection-based methods

- Use visual object detectors
- Sum up the occurrences of detected objects
- Limited by occlusions and background cluttering

Traditional approaches

Detection-based methods

- Use visual object detectors
- Sum up the occurrences of detected objects
- Limited by occlusions and background cluttering

Regression-based methods

- They try to learn a relation from image features to people count
- A linear mapping is improbable and difficult to learn (Pham et al.)
- Random forest regression methods can learn a non-linear mapping between local features and density maps

Traditional approaches

Detection-based methods

- Use visual object detectors
- Sum up the occurrences of detected objects
- Limited by occlusions and background cluttering

Regression-based methods

- They try to learn a relation from image features to people count
- A linear mapping is improbable and difficult to learn (Pham et al.)
- Random forest regression methods can learn a non-linear mapping between local features and density maps

More on density maps later

CNN-based approaches

Many methods have been tried

- CNN that maps each image frame to its crowd-count value (Wang et al.)
- Train on crowd density and crowd count alternatively (Zhang et al.)
- These approaches didn't generalize well with scale variations
- They usually require too many epochs and parameters

More recently:

- Cao et al. proposed SANet, an encoder-decoder network for accurate crowd counting
- Li et al. proposed combining VGG-16 with dilated convolutional layers in CSRNet
- Density levels and high-resolution density maps have been exploited by Sindagi et al.

CNN-based approaches

Many methods have been tried

- CNN that maps each image frame to its crowd-count value (Wang et al.)
- Train on crowd density and crowd count alternatively (Zhang et al.)
- These approaches didn't generalize well with scale variations
- They usually require too many epochs and parameters

More recently:

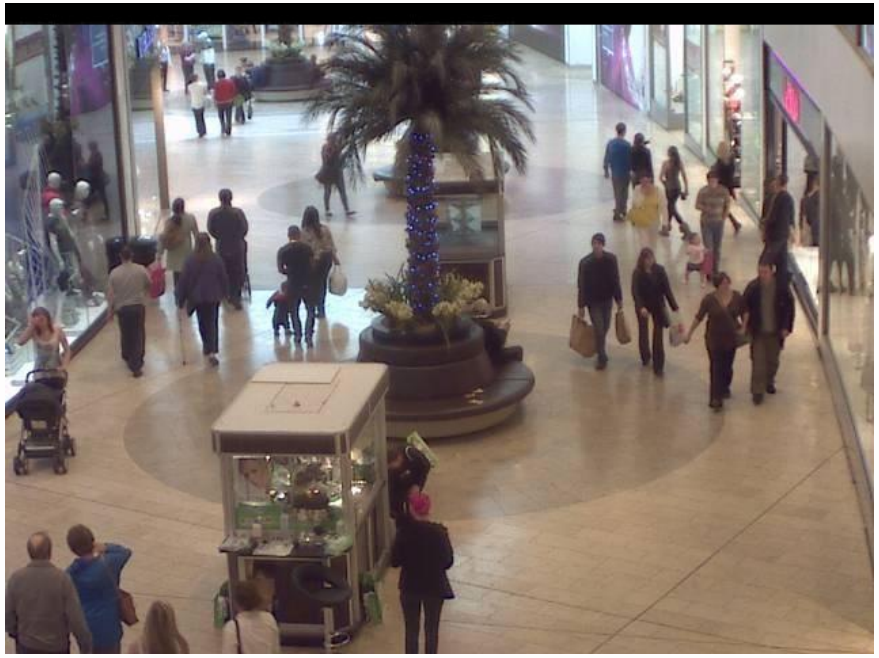
- Cao et al. proposed SANet, an encoder-decoder network for accurate crowd counting
- Li et al. proposed combining VGG-16 with dilated convolutional layers in CSRNet
- Density levels and high-resolution density maps have been exploited by Sindagi et al.

No clear winner...

Datasets

Dataset: Mall

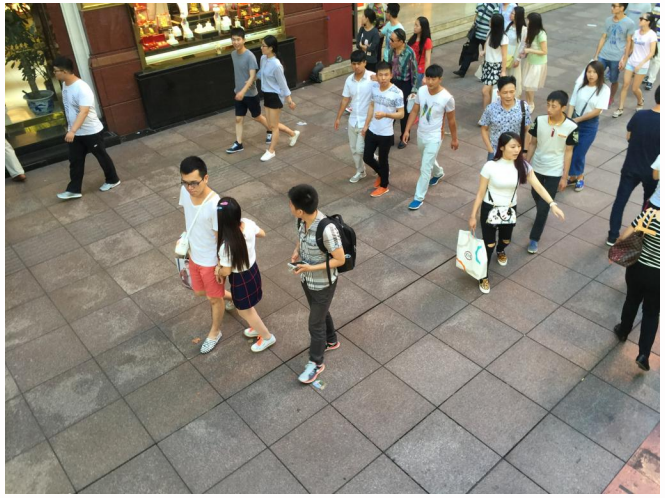
- 2000 RGB frames
- Fixed frame size: 640x480 pixels
- (😔) Same scene recorded from a security camera in a Mall
- (😔) The top of every image has a black band



Dataset: Shanghai Tech

- Actually 2 datasets: Part A and Part B
  - Part A has 700 images
  - Part B has 498 images
- 1198 RGB frames in total

- ~330.000 head coordinates
- Different frame sizes
- (😋) Different viewpoints
- (😋) Various conditions



Preprocessing

Mall dataset:

- Remove the black top 16 pixels from images
- Random data split (80/20)

Shanghai Tech datasets:

- Extract ground truth head-point coordinates from `*.mat` files to `numpy` files

Both datasets:

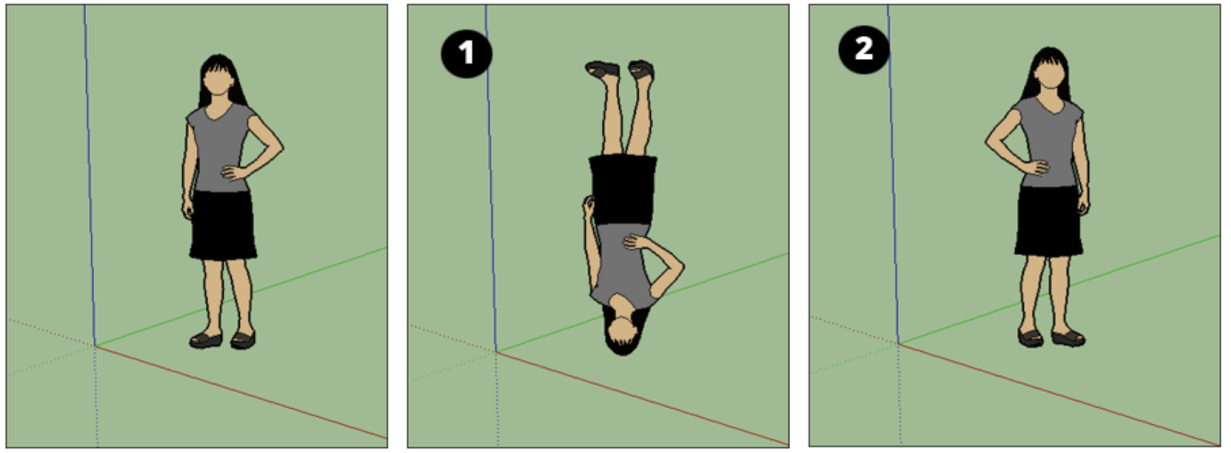
- Extract ground truth count from `*.mat` files to `CSV` files

Data augmentation

- The datasets are very small
- Hard to find datasets with head points, they require a lot of manual effort
- Provide scale, rotation and flips invariance
- (equivariance for density maps as we will see)

We then applied to each image batch:

- Random flips (horizontal)
- Random rotations
- Random resizing and rescaling



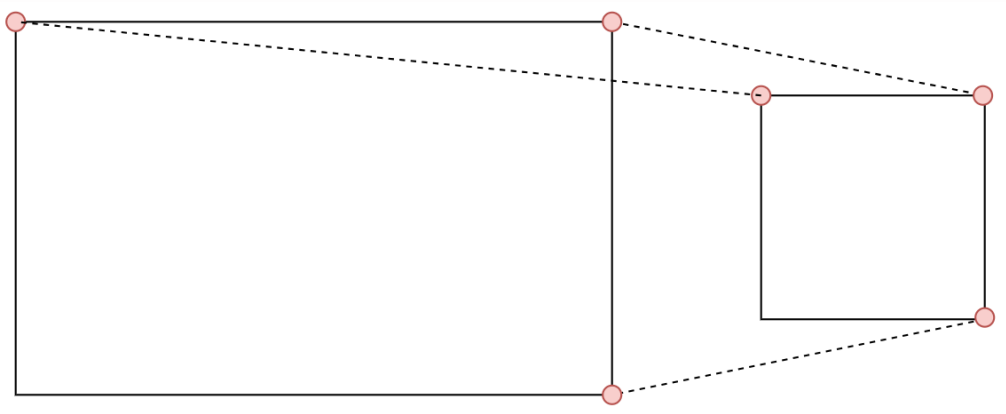
How do we adapt the ground truth to the resized images?

Affine transformations (1)

Given sets of coordinates (x_i, y_i) in the original image and (x'_i, y'_i) in the resized image:

- Infer the transformation matrix that maps (x_i, y_i) to (x'_i, y'_i) using 3 extreme points
- Use `map_matrix` to transform the ground truth points to the new scale

$$\begin{bmatrix} x'_i \\ y'_i \end{bmatrix} = \texttt{map\_matrix} \cdot \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

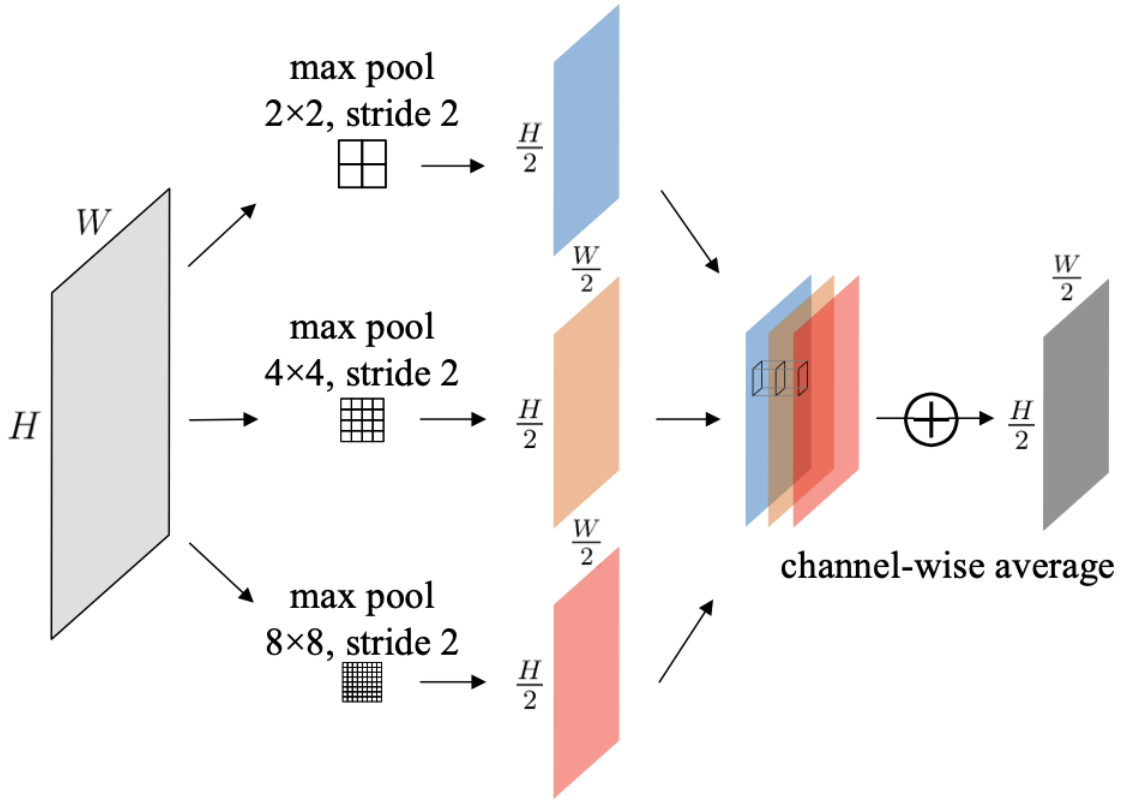


Affine transformations (2)

A first attempt

- CNN with Multi-Kernel pooling
- Output layer: single neuron
- Predicts automatically the crowd count

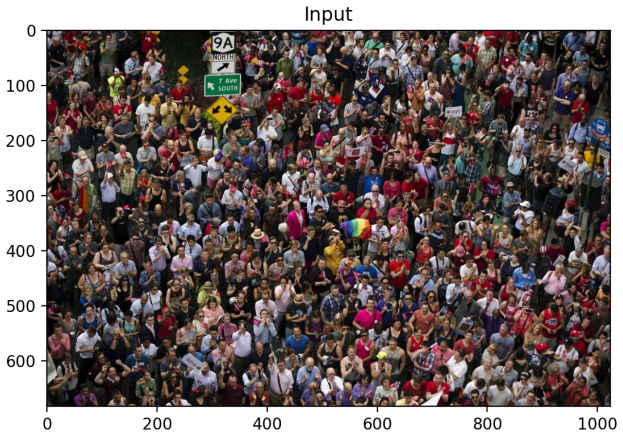
However: **overfitting**

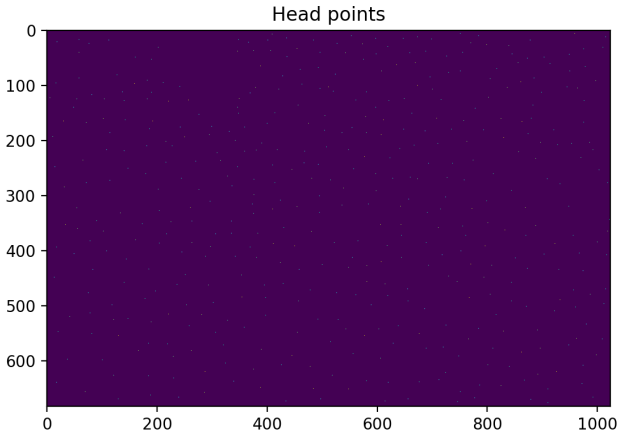


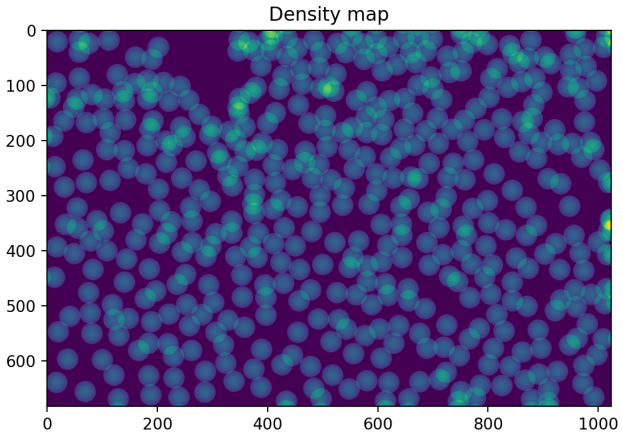
Methods & Experiments

Density maps

- Exploit all ground-truth information
- Convert head points into a map
- Create an image with head points
- Convolution with a Gaussian Kernel







Density maps

Gaussian kernel:

- Size: 91x91 pixels
- $\sigma = 15$
- All values below 0.0003 are set to 0
- Normalization: sum to 1

1/273

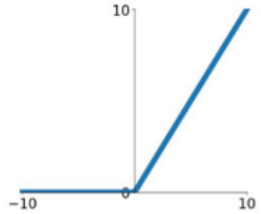
1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

Architecture

- Transfer learning: VGG16 pre-trained on ImageNet
- Removed fully connected layers
- Upsampling through transposed convolutional layers
- Convolution Kernel size: 3×3, stride 1×1
- Activation: ReLU

ReLU

$\max(0, x)$

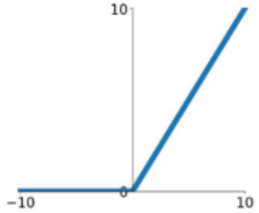


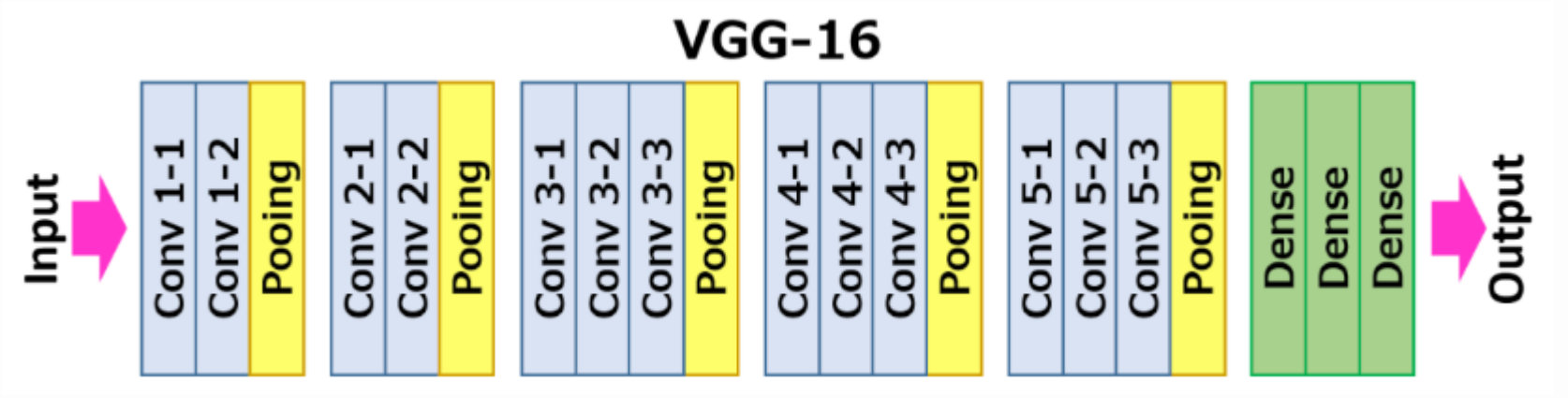
Architecture

- Transfer learning: VGG16 pre-trained on ImageNet
- Removed fully connected layers
- Upsampling through transposed convolutional layers
- Convolution Kernel size: 3×3, stride 1×1
- Activation: ReLU

ReLU

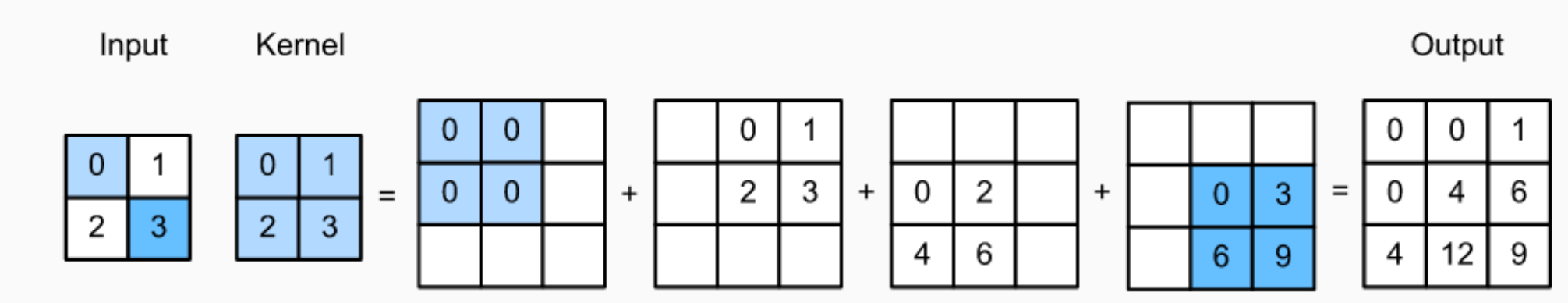
$\max(0, x)$



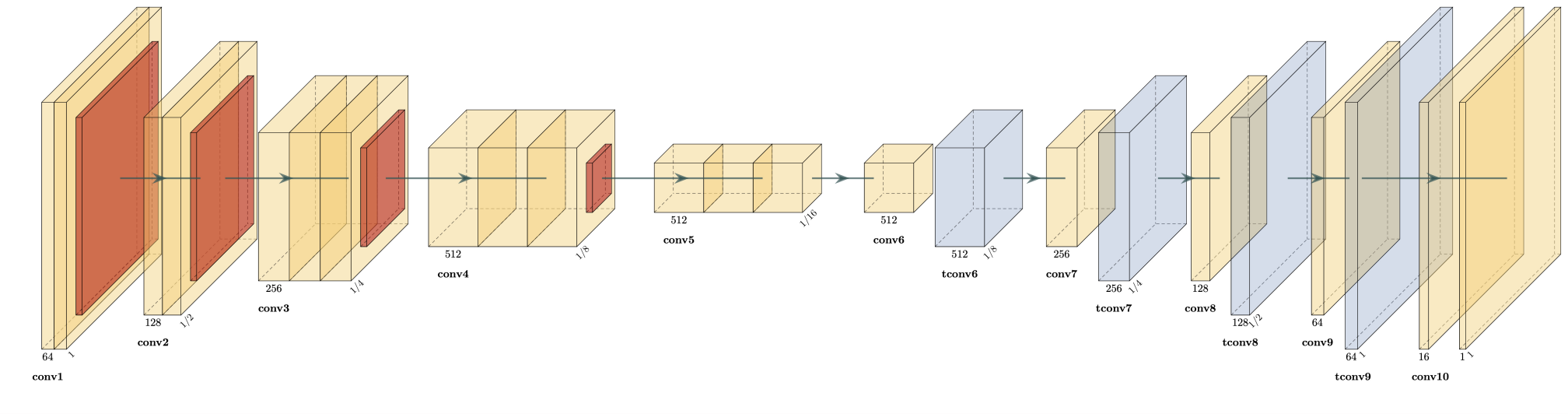


Transpose convolution

- Upsampled convolution
- Kernel size: 2×2, stride 2×2
- Activation: ReLU



Architecture

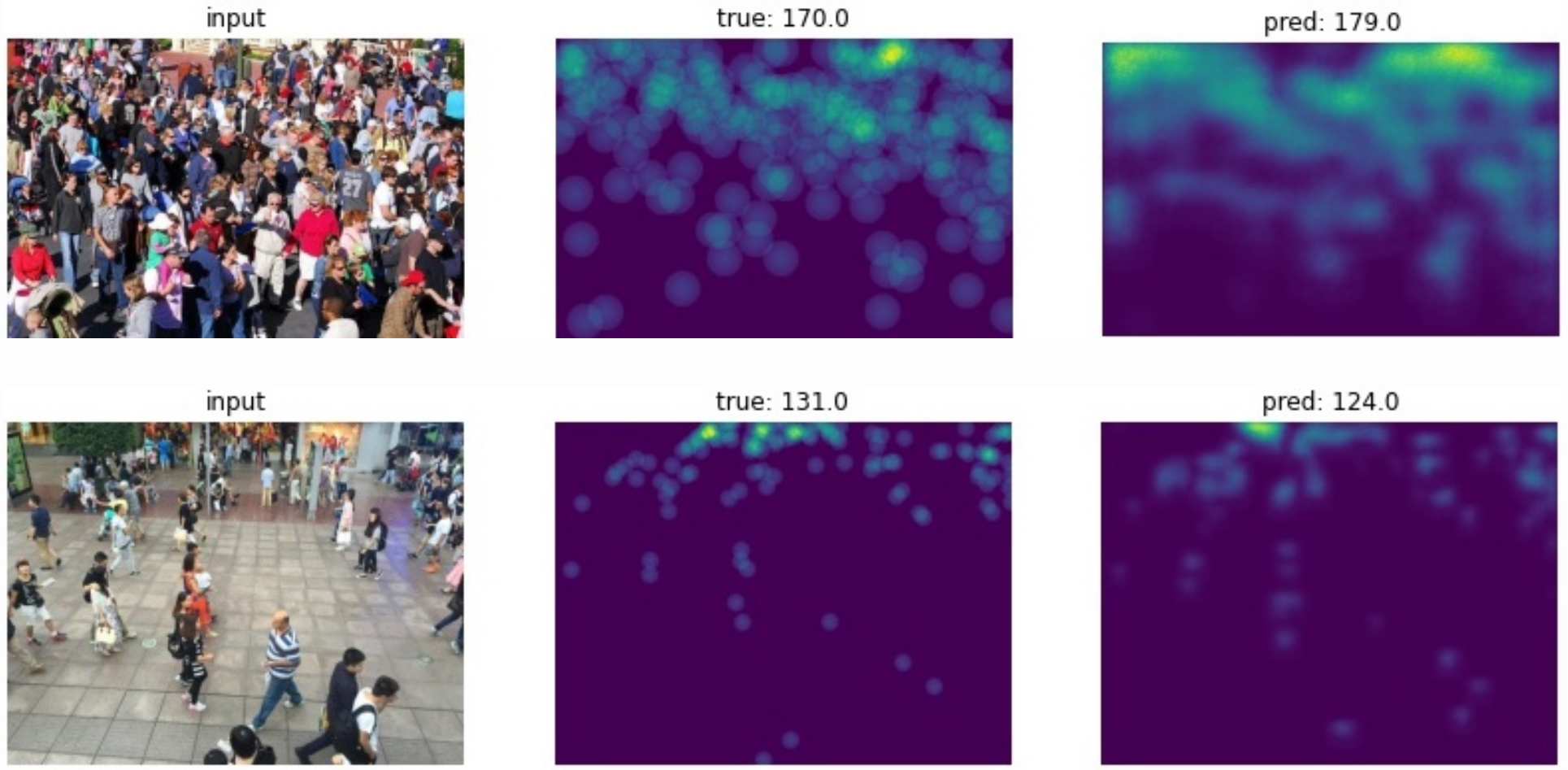


Training

- Batch size: 1
- Online data augmentation
- Learning rate: from $1e-5$ to $1e-7$
- Optimizer: Adam
- Alternating between Shanghai Tech A and B every 2 epochs
- Loss: euclidean distance $L_c = (y - \hat{y})^2$
- Best validation loss after 126 epochs

Prediction

Examples:



Metrics

$\hat{z}_i$: prediction
z_i: ground truth

- $\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N} |\hat{z}_i - z_i|$
- $\mathrm{MSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} (\hat{z}_i - z_i)^2}$
- $\mathrm{MAPE} = \frac{1}{N}\sum_{i=1}^{N} \frac{|\hat{z}_i - z_i|}{z_i} * 100$

Results

	Part A			Part B		
Method	**MAE**	**MSE**	**MAPE**	**MAE**	**MSE**	**MAPE**
Zhang et al. (Cross-scene CNN, 2015)	181.8	277.7		32.0	49.8	
Zhang et al. (Multi-column CNN, 2016)	110.2	173.2		26.4	41.3	
Babu Sam et al. (Switching CNN, 2017)	90.4	135.0		21.6	33.4	
Sindagi et al. (CP-CNN, 2017)	73.6	106.4		20.1	30.1	
Liu et al. (2018)	72.0	106.6		14.4	23.8	
Liu et al. (2018)	73.6	112.0		13.7	21.4	
Ours	92.8	148.2	28.2%	16.9	28.1	22.8%

Results

	Part A			Part B		
Method	**MAE**	**MSE**	**MAPE**	**MAE**	**MSE**	**MAPE**
Zhang et al. (Cross-scene CNN, 2015)	181.8	277.7		32.0	49.8	
Zhang et al. (Multi-column CNN, 2016)	110.2	173.2		26.4	41.3	
Babu Sam et al. (Switching CNN, 2017)	90.4	135.0		21.6	33.4	
Sindagi et al. (CP-CNN, 2017)	73.6	106.4		20.1	30.1	
Liu et al. (2018)	72.0	106.6		14.4	23.8	
Liu et al. (2018)	73.6	112.0		13.7	21.4	
Ours	92.8	148.2	28.2%	16.9	28.1	22.8%

Resources:

- Google Cloud VM
- 8CPUs, 52 GB RAM
- 1 NVIDIA Tesla T4

Conclusions

Our contribution

- A novel CNN for high-resolution density maps prediction
- A novel approach for data augmentation
- A full explanation on how to predict density maps

What we learned

- Hands-on experience with a CV task
- Preprocessing
- Transfer learning
- Importance of data
- How to design a model
- Google Cloud Platform
- Experience with state-of-the-art research paper

Future work

- More accurate loss function
- Adaptive gaussian kernel
- Different baseline networks (SqueezeNet, Xception)
- Dilated convolution
- Main problem: small dataset

Thank you for your attention

Questions?