Practical Approach to Evacuation Planning Via Network Flow and Deep Learning

Akira Tanaka※ Nozomi Hata※

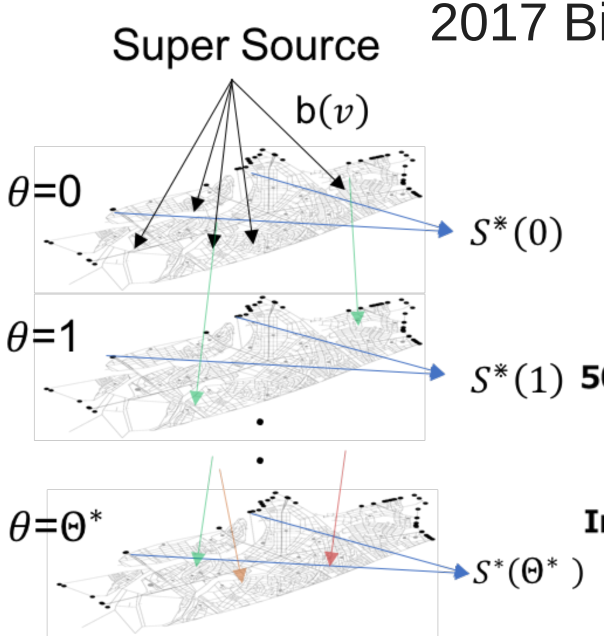
Nariaki Tateiwa※ Katsuki Fujisawa †

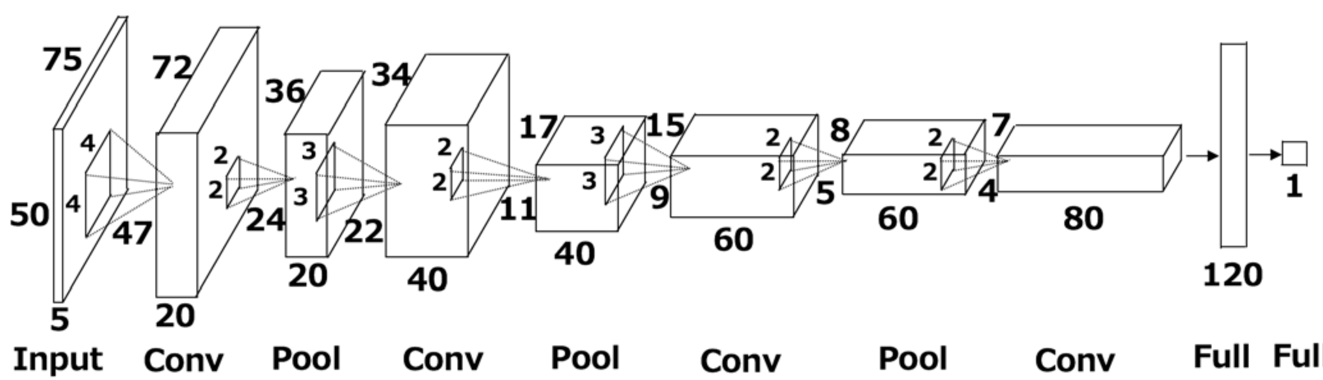
※ Graduate School of Mathematics, Kyushu University

† Institute of Mathematics for industry, Kyushu University

December 11, 2017

2017 BigGraphs Workshop at IEEE BigData17





Evacuation Map

Yodogawa Area in Osaka City

#nodes : 2,933

#edges : 8,924

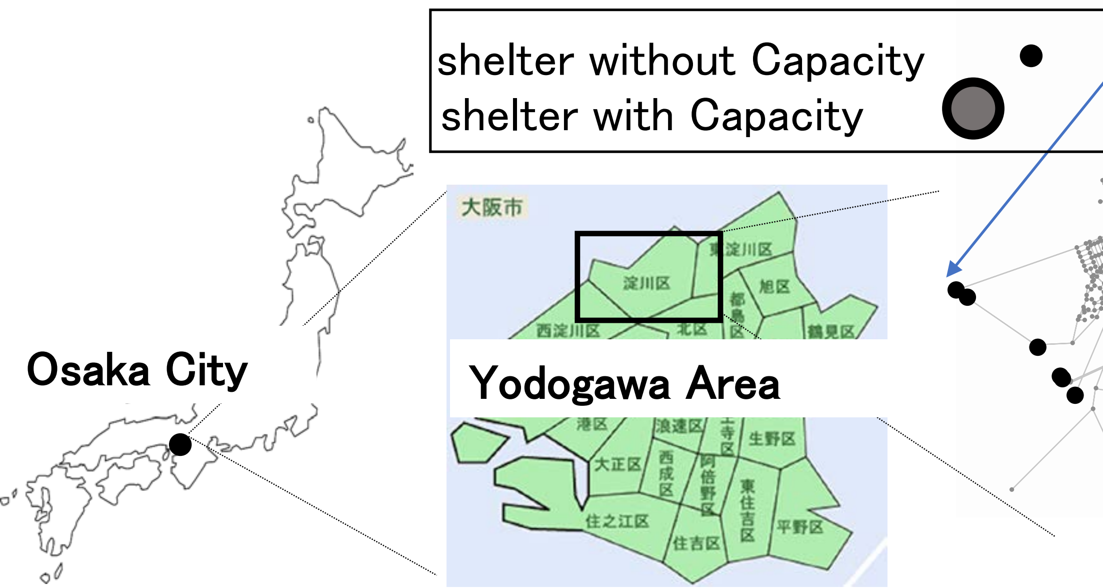
#evacuees : 50,000 ~ 80,000 ← Based on midnight population

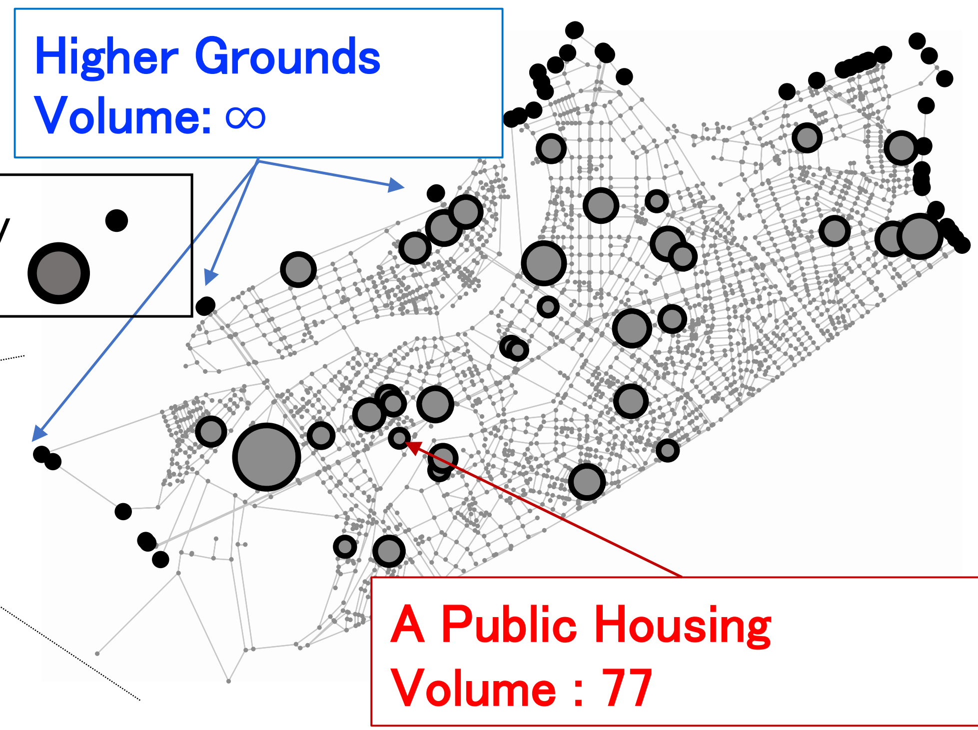
#shelters

- with finite capacity : 36 (total volume:35,549)
- with infinite capacity : 50 (each volume:∞)

Higher Grounds

Tall and Strong Buildings

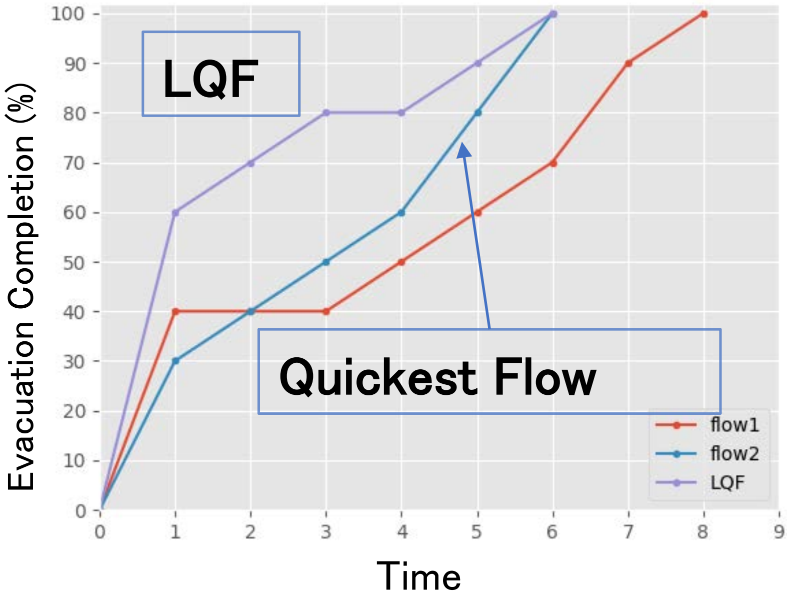




Lexicographically Quickest Flow(LQF)

Definition

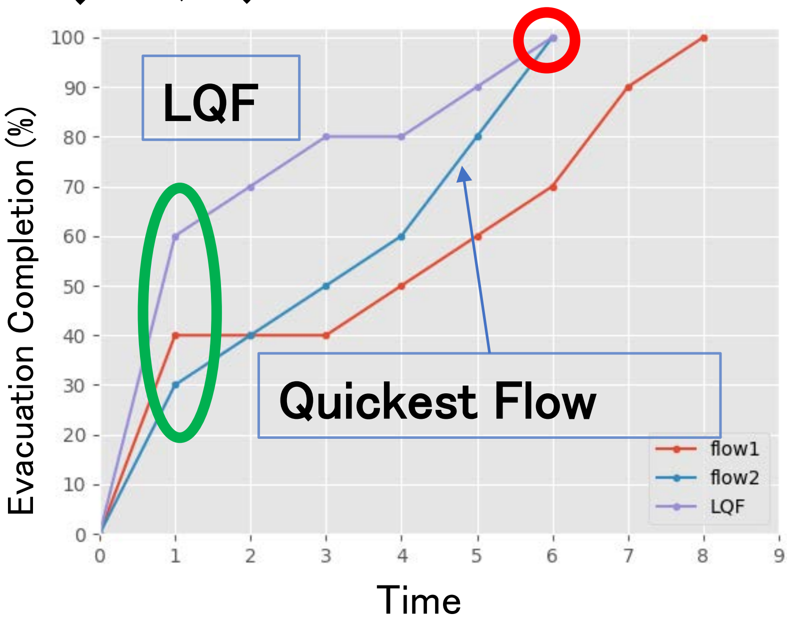
- Minimizes the evacuation time Θ^*
- Greedily maximize the cumulative number of evacuees who have already completed evacuation at every time θ in the order of $\theta = 1,2,..,\Theta^*$



Lexicographically Quickest Flow(LQF)

Definition

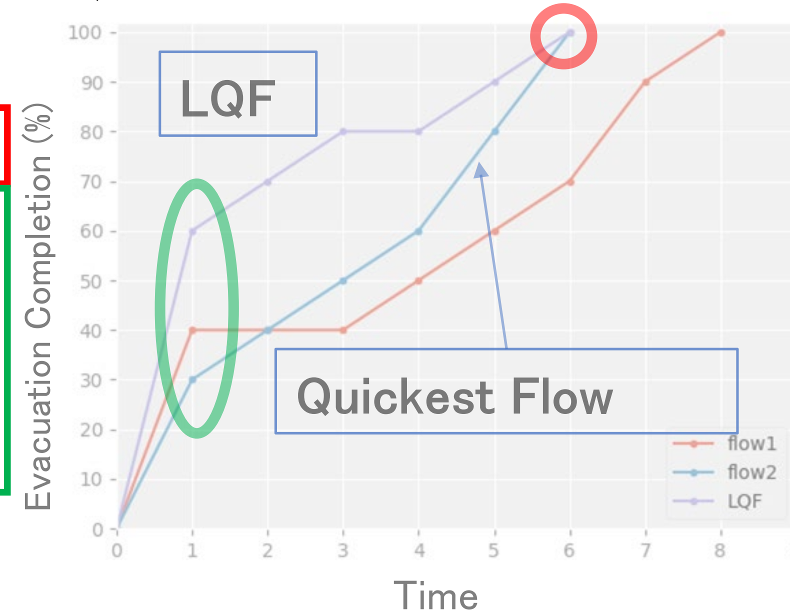
- Minimizes the evacuation time Θ^*
- Greedily maximize the cumulative number of evacuees who have already completed evacuation at every time θ in the order of $\theta = 1,2,..,\Theta^*$



Lexicographically Quickest Flow(LQF)

Definition

- Minimizes the evacuation time Θ^*
- Greedily maximize the cumulative number of evacuees who have already completed evacuation at every time θ in the order of $\theta = 1,2,..,\Theta^*$



Practical Property

- The LQF also has a high efficiency with respect to total pedestrian flow of all evacuees to the refuges
- To check this property, we decomposed the LQF into cyclic flow and cycle-free flow by solving the QP below:

Flow f

Cyclic Flow f_c
(only 2 cycles remained)

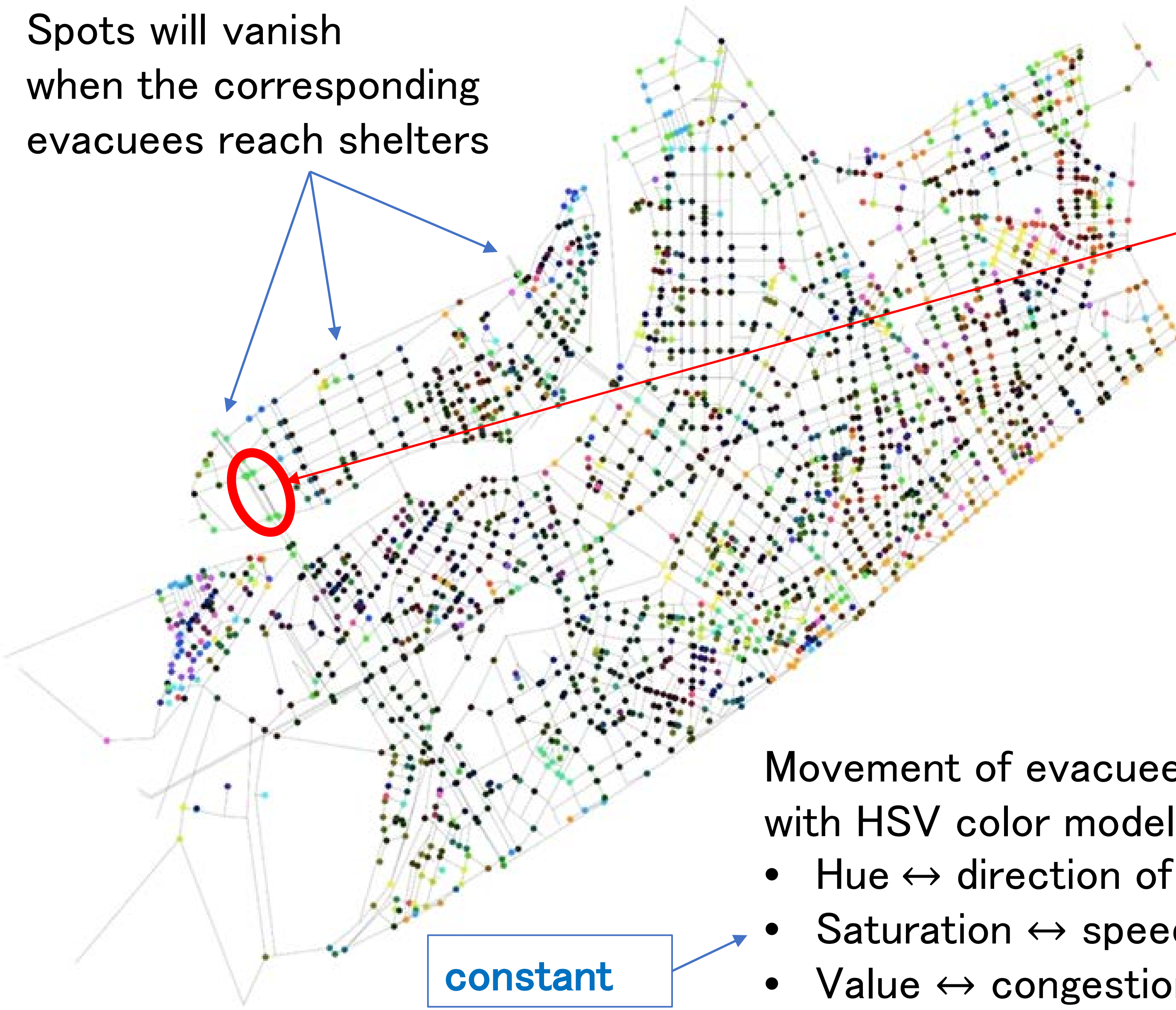
$$
\begin{aligned}
\text{minimize} \quad & \sum_{e\in A}(f(e)-f_c(e))^2 \\
\text{subject to} \quad & f_c \in \mathcal{F}_c \\
& 0 \le f_c(e) \le f(e) \qquad (\forall e \in A \text{ s.t. } f(e) \ge 0) \\
& f(e) \le f_c(e) \le 0 \qquad (\forall e \in A \text{ s.t. } f(e) \le 0)
\end{aligned}
$$

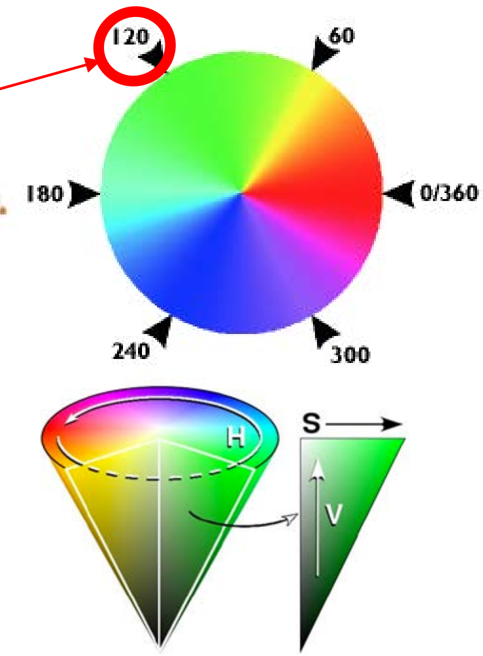
where

- $f(e)$ is the total refuges that passed the arc e
- $A:$ the set of arcs
- $\mathcal{F}_c := \{f : A \to \mathbb{R} \mid f \text{ is a cyclic flow}\}$

Visualization of LQF

Spots will vanish
when the corresponding
evacuees reach shelters





Movement of evacuees are expressed with HSV color model

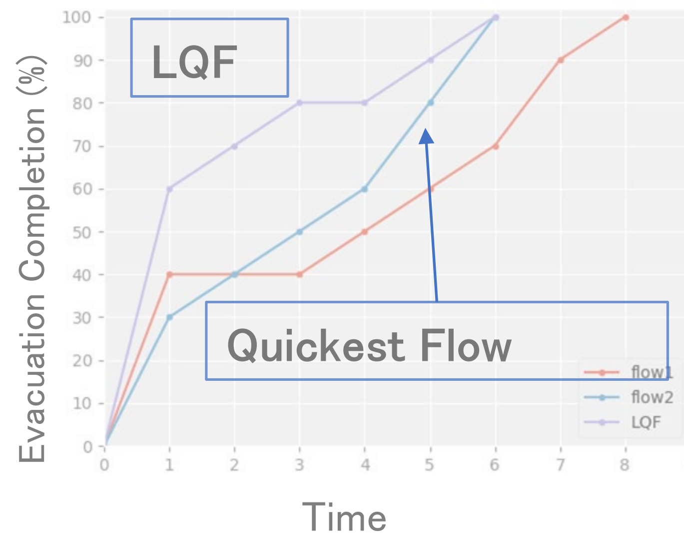
- Hue ↔ direction of movement
- Saturation ↔ speed of movement
- Value ↔ congestion of arc

constant

Lexicographically Quickest Flow (LQF)

Pros(practical property)

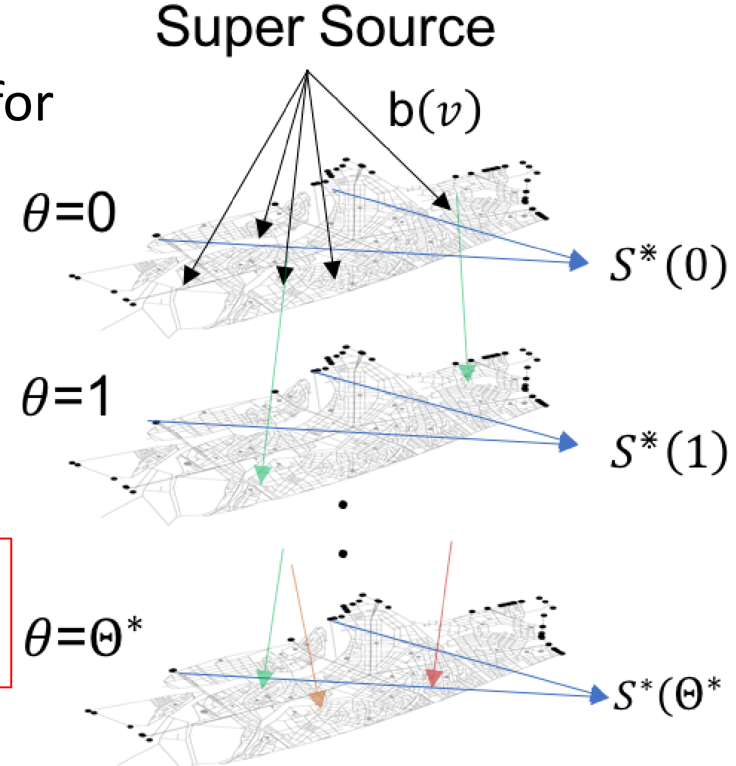
- Minimizes the evacuation time Θ^*
- Greedily maximize the cumulative number of evacuees
  →More people reach shelters in the early stage
- There are few unnecessary movements



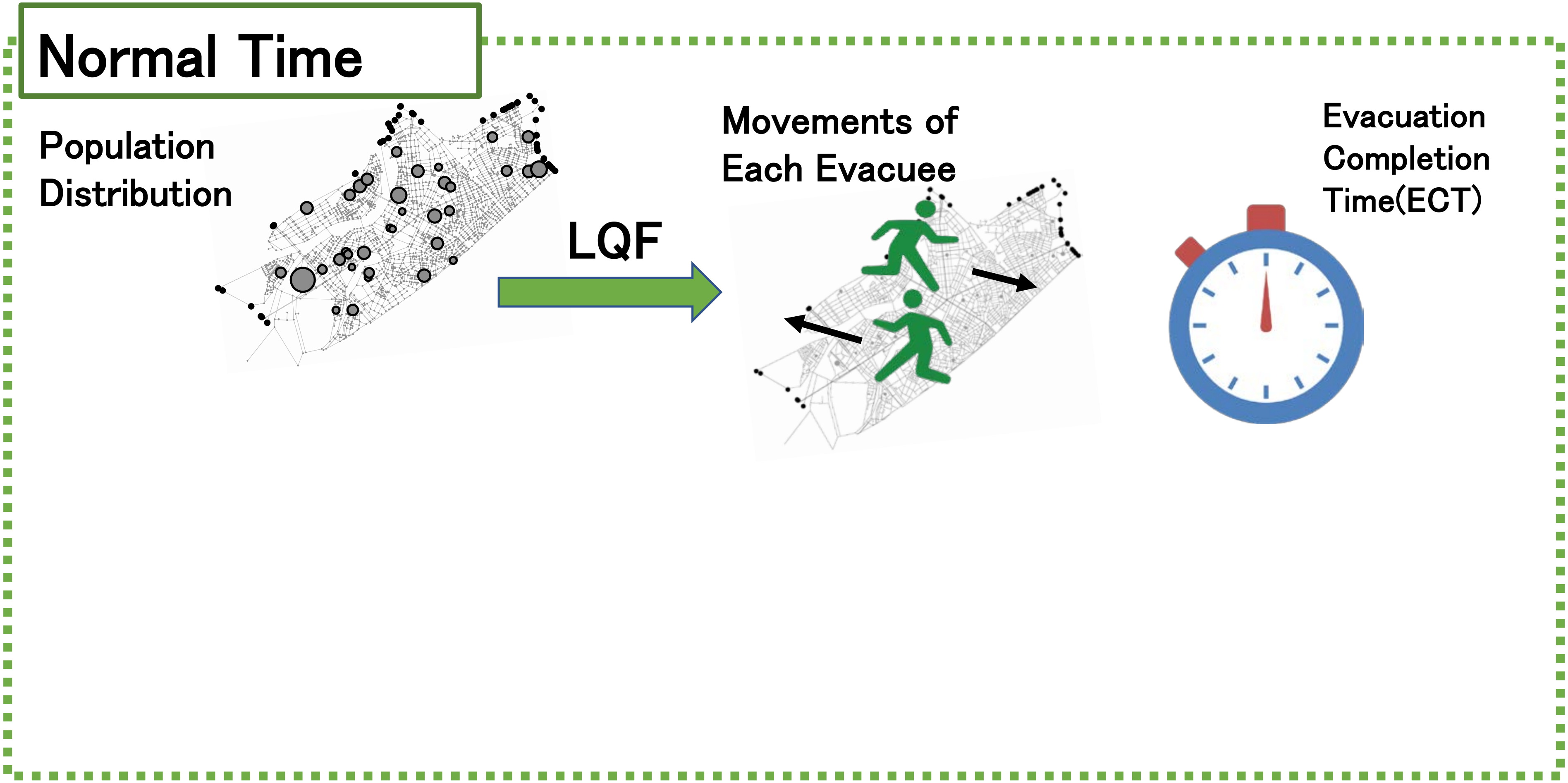
Cons

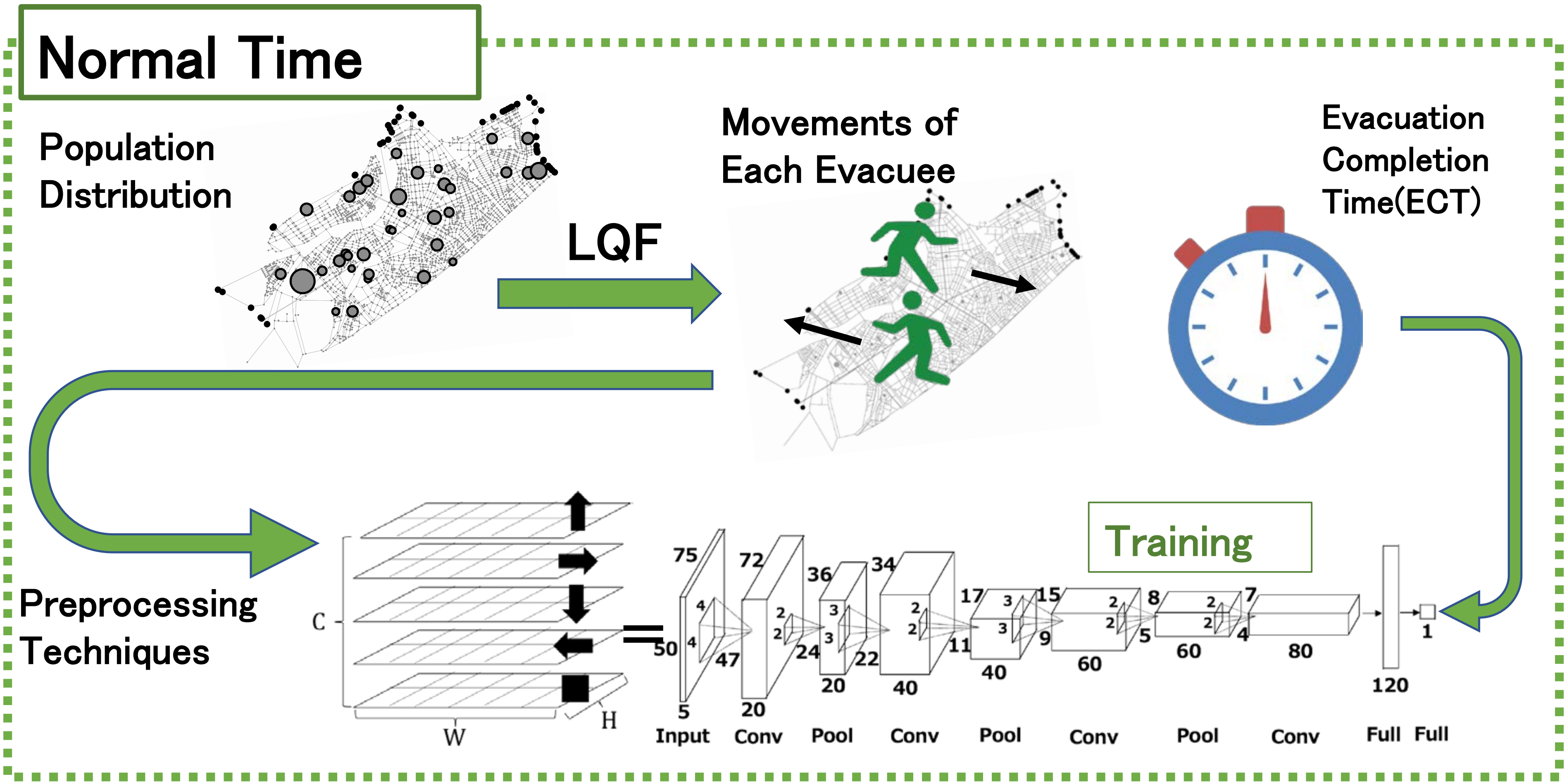
- Not practical(7h for the target area)
  - Compute the maximum flow repeatedly for the time-expanted graph.
  - Size of time-expanded graph is huge

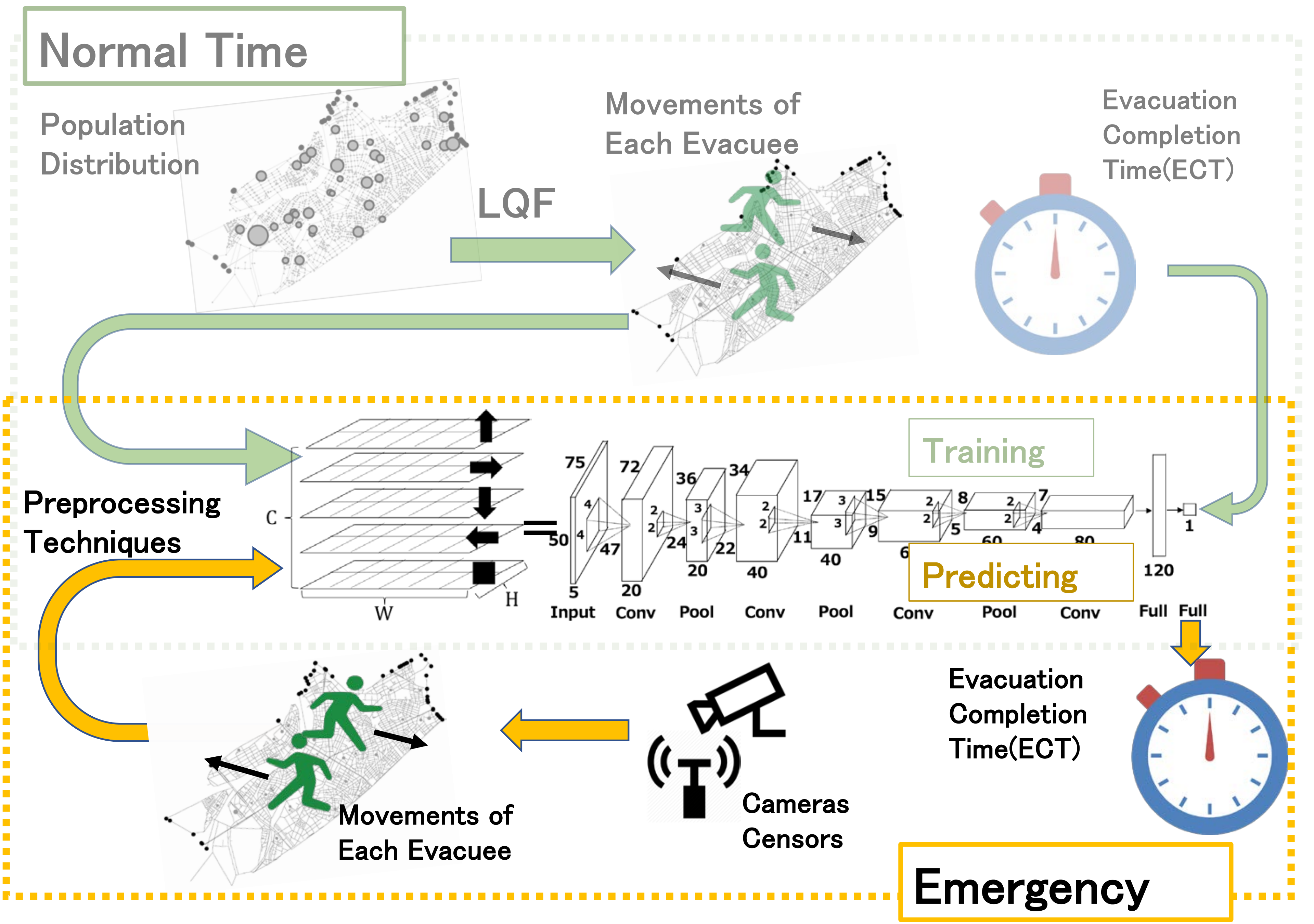
	Original	Time-Expanded($\theta = 2,824$)
#nodes	2,933	8,300,000
#edges	8,924	32,800,000



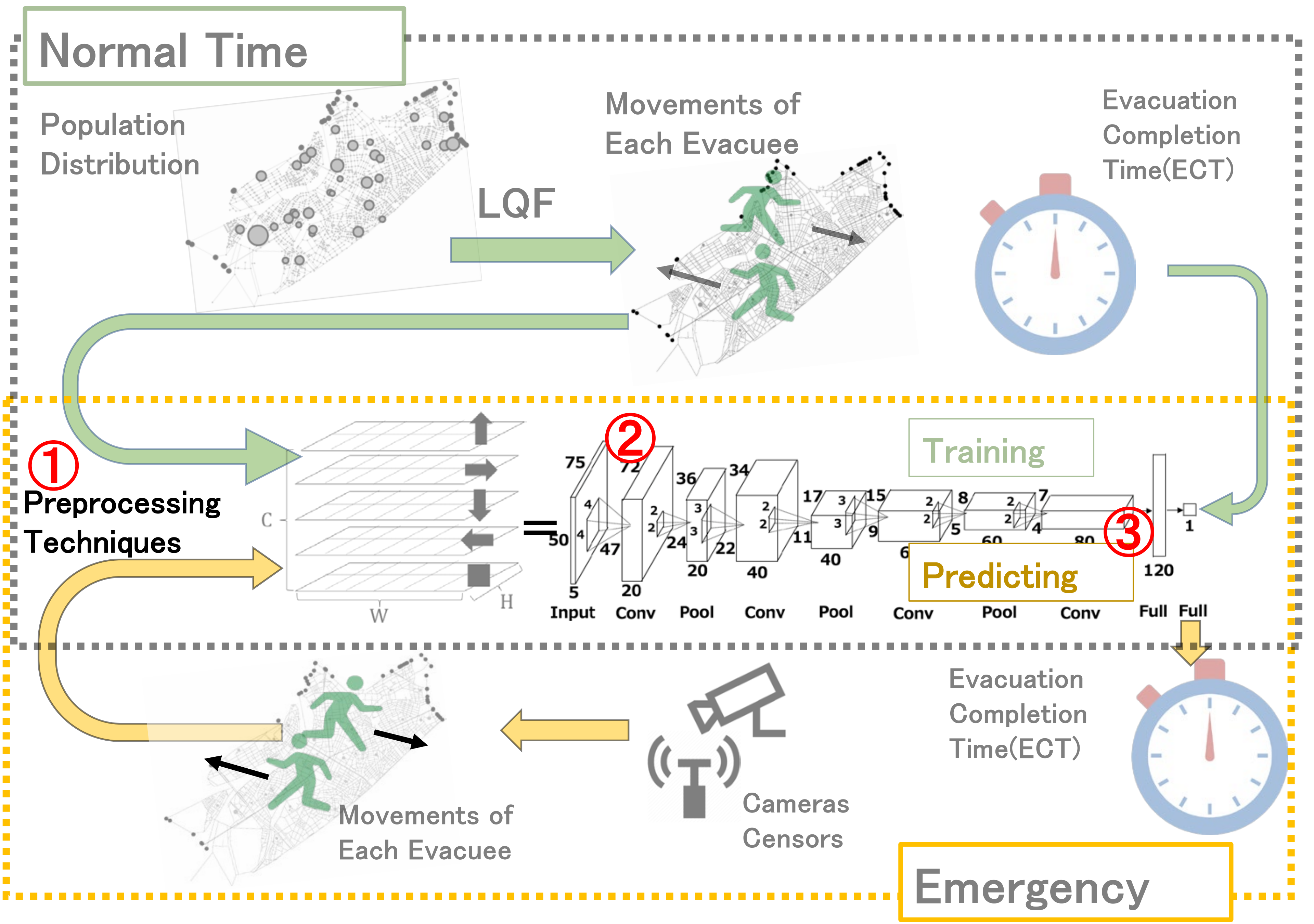
To utilize practical property of LQF, we combine it with Deep Learning


Normal Time
Population Distribution
LQF
Movements of Each Evacuee
Evacuation Completion Time(ECT)


Normal Time
Population Distribution
LQF
Movements of Each Evacuee
Evacuation Completion Time(ECT)
Preprocessing Techniques
Training
C
W
H
Input
Conv
Pool
Conv
Pool
Conv
Pool
Conv
Full
Full

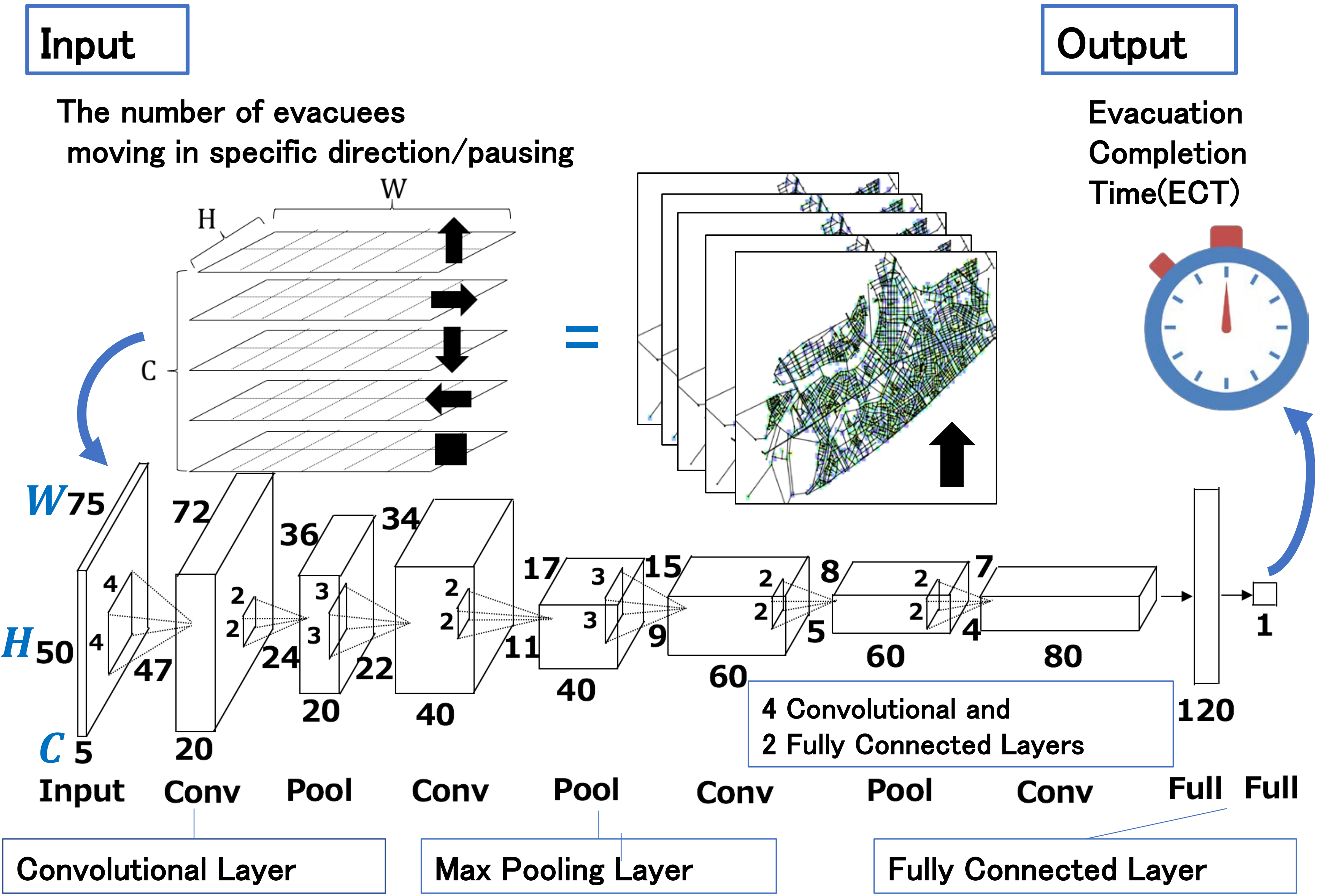


Normal Time
Population Distribution
LQF
Movements of Each Evacuee
Evacuation Completion Time(ECT)
Preprocessing Techniques
C
W
H
Training
Predicting
Input
Conv
Pool
Conv
Pool
Conv
Pool
Conv
Full
Full
Evacuation Completion Time(ECT)
Cameras Censors
Movements of Each Evacuee
Emergency

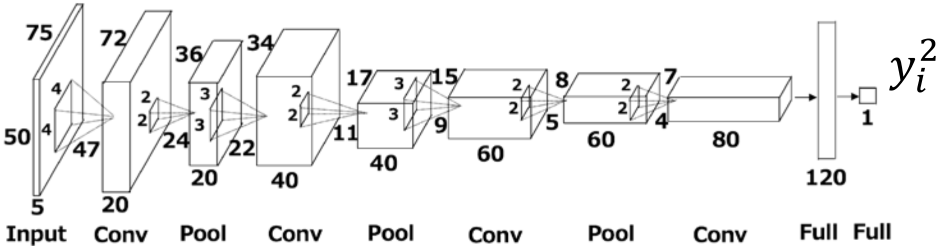


Normal Time
Population Distribution
LQF
Movements of Each Evacuee
Evacuation Completion Time(ECT)
①
Preprocessing Techniques
②
Training
Predicting
③
Input
Conv
Pool
Conv
Pool
Conv
Pool
Conv
Full
Full
Movements of Each Evacuee
Cameras Censors
Evacuation Completion Time(ECT)
Emergency

Convolutional Neural Network(CNN)



Optimizer and Output of CNN

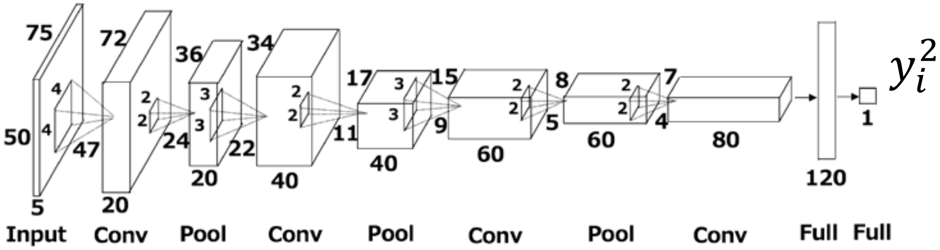


Euclidean Loss (cost function)

$$\frac{1}{2N}\sum_{i=1}^{N}\left\|y_i^1 - y_i^2\right\|^2$$

y_i^1:Accuare output(LQF_T)
y_i^2:Predicting output
N:data size

Optimizer and Output of CNN



Euclidean Loss (cost function)

$$\frac{1}{2N}\sum_{i=1}^{N}\left\|y_i^1 - y_i^2\right\|^2$$

y_i^1:Accuare output(LQF_T)
y_i^2:Predicting output
N:data size

Adam(optimizer)

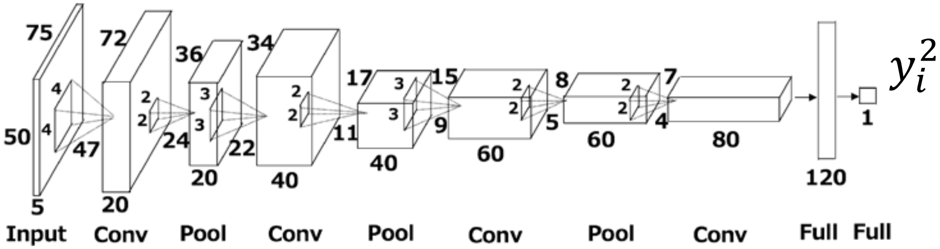
- First-order gradient-based optimization

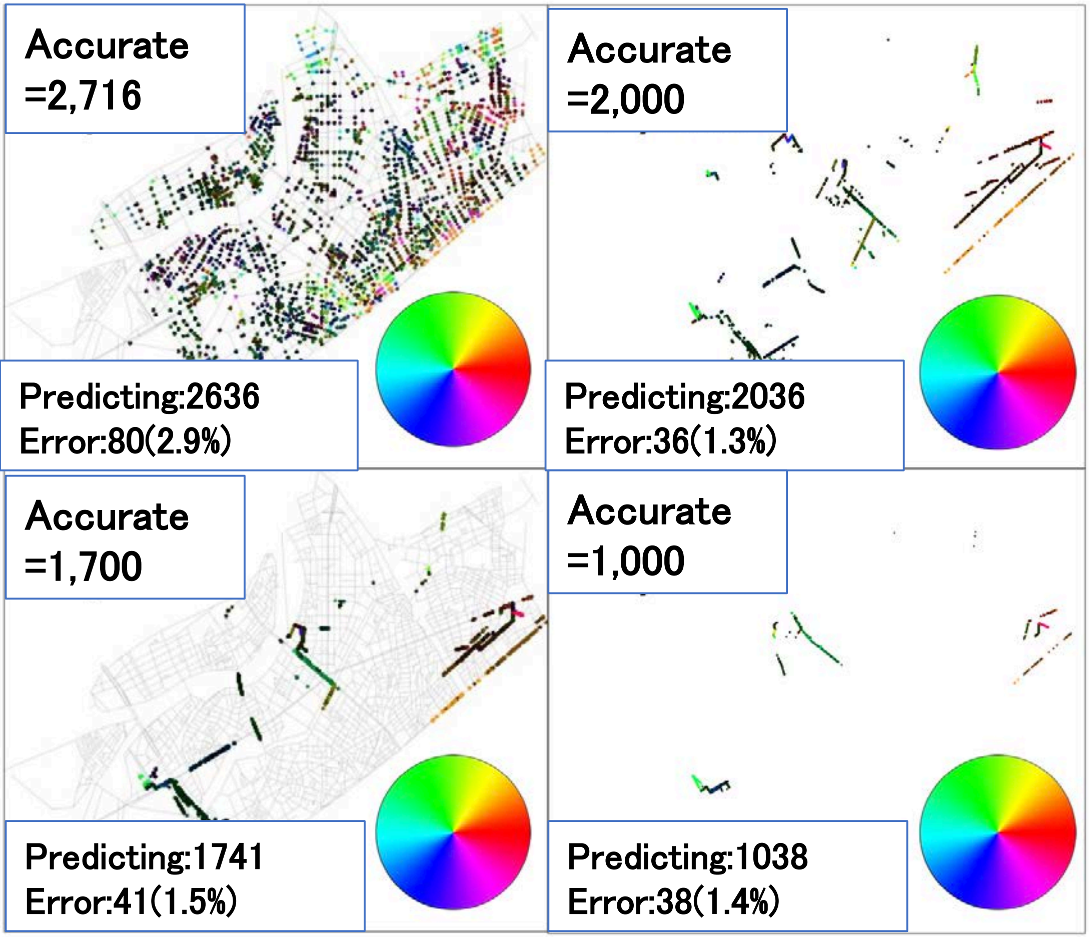
→ feasible to compute in practice for high-dimensional data

- Storing exponentially decaying average of past gradient and squared gradient

→Large updates for infrequent and smaller updates for frequent
→accelerate learning

Optimizer and Output of CNN





Error := |Accurate − Predicting| (Error / 2,751)
Note: 2,751 is the average evacuation time

Euclidean Loss (cost function)

$$\frac{1}{2N}\sum_{i=1}^{N}\left\|y_i^1 - y_i^2\right\|^2$$

y_i^1:Accuare output(LQF_T)
y_i^2:Predicting output
N:data size

Adam(optimizer)

- First-order gradient-based optimization

→ feasible to compute in practice for high-dimensional data

- Storing exponentially decaying average of past gradient and squared gradient

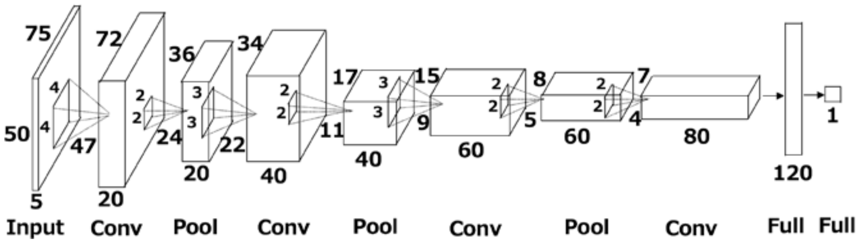
→Large updates for infrequent and smaller updates for frequent
→accelerate learning

Results of Our Model

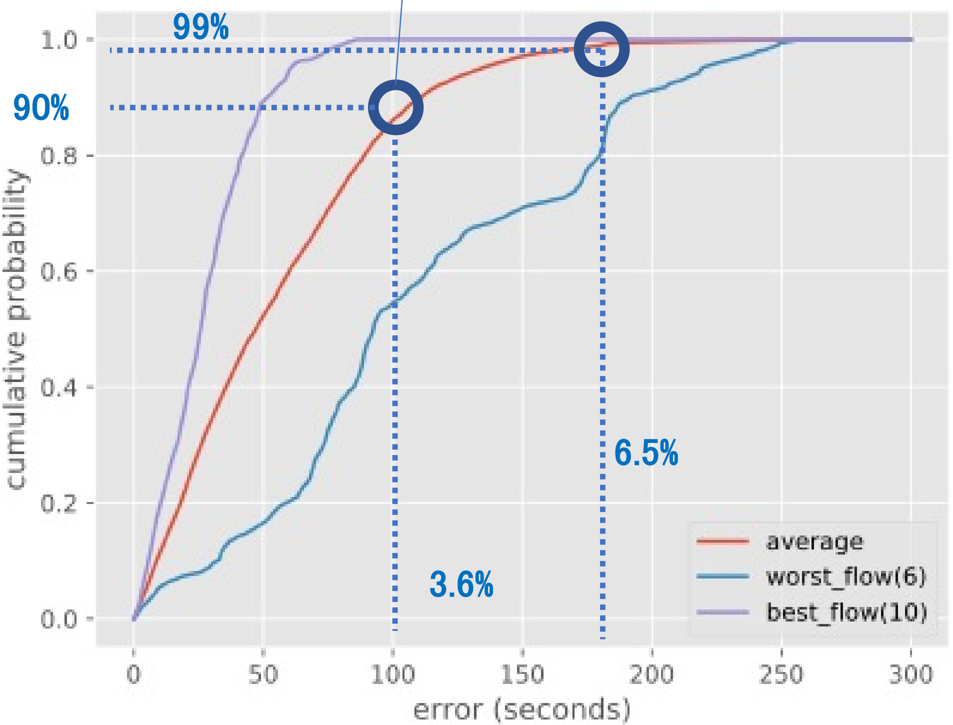
Data

#Training Data: 43,200

#Test Data: 5,400



90% probability that we make prediction within 3.6% error



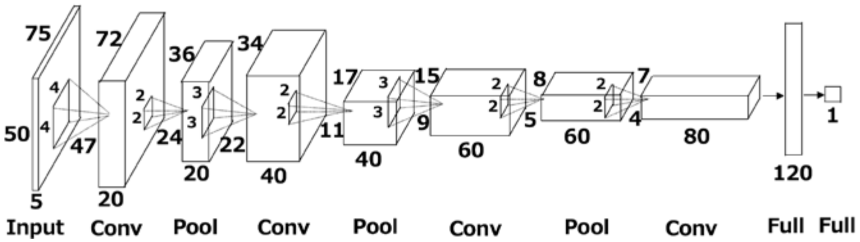
Cumulative Probabilities

Results of Our Model

Data

#Training Data: 43,200

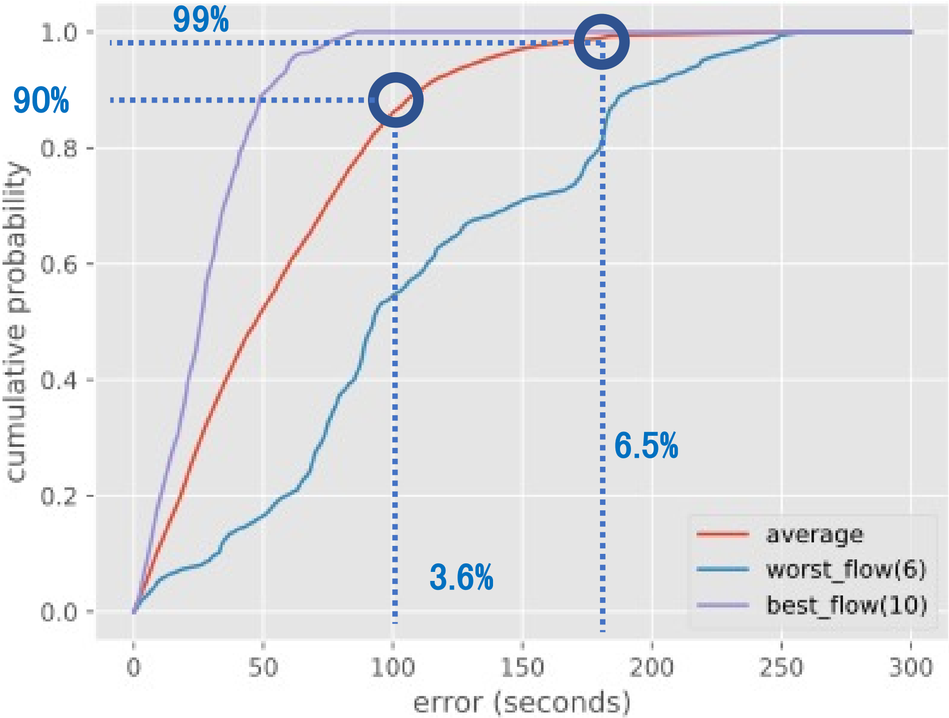
#Test Data: 5,400



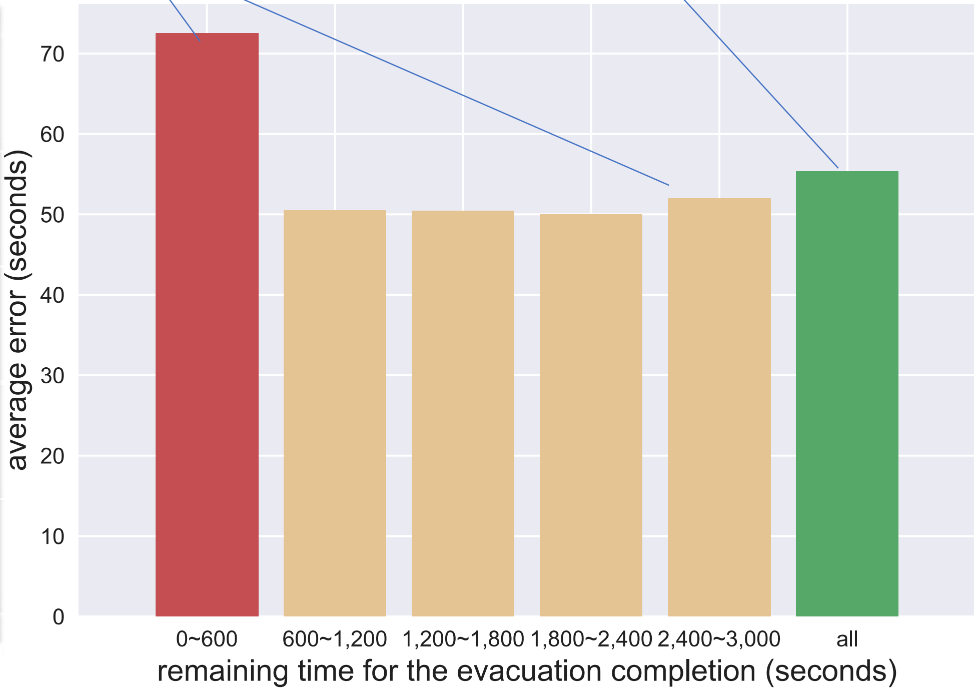
90% probability that we make prediction within 3.6% error

Average error(55s 2%) is quite a small

Prediction for the early stage is better than that for the final stage



Cumulative Probabilities



Average Error in Several Stages of Evacuation

Summary

Previous Model(LQF)

Not practical(7h)

- Compute max-flow repeatedly for time-expanded graph
- Time-expanded graph is huge

Proposed Model

Practical (less than 1s)

- Combing LQF and CNN
- Predict evacuation completion time immediately(less than 1s) and almost accurately(2%)