Bayesian approaches and ML techniques in multisource fusion systems

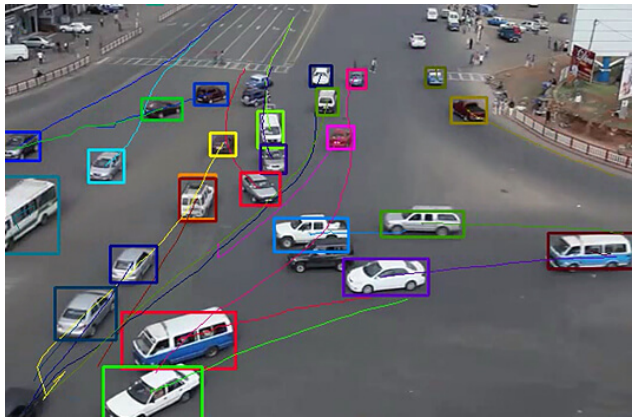
Emanuel Aldea

Data Analysis Group, SATIE Laboratory, Université Paris-Saclay

Outline

Outline

Filtering application : object tracking

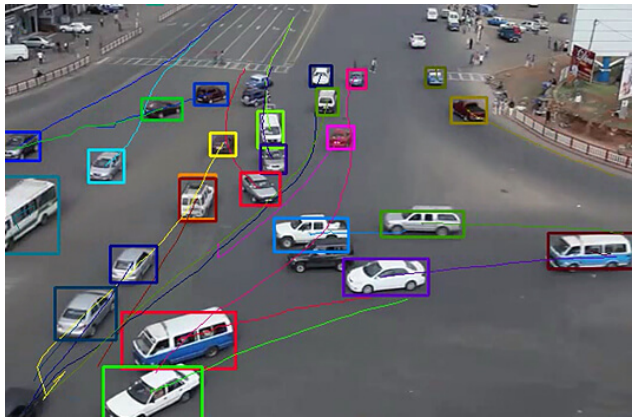


A very popular and useful application in computer vision, with significant interest in human-robot interaction and autonomous systems, surveillance and traffic monitoring, human activity recognition, biomedical imaging etc.

Main objective

- Track the object(s) of interest as they move around during a video
- The state contains the current location, and typically information about scale, appearance and dynamics
- Not all trackers are equal : computational cost, handling occlusions, handling complex dynamics, handling the presence of visually similar targets, handling multiple target interactions

Filtering application : object tracking

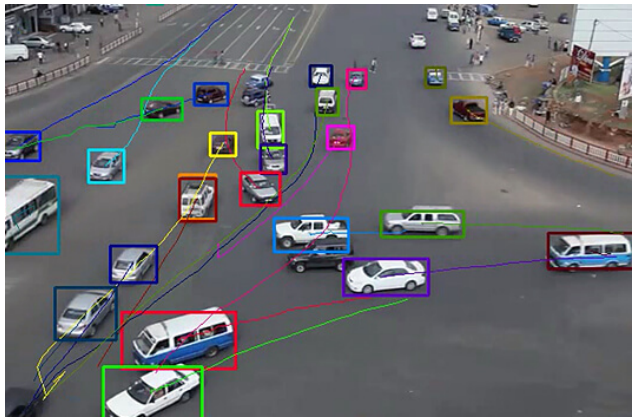


A very popular and useful application in computer vision, with significant interest in human-robot interaction and autonomous systems, surveillance and traffic monitoring, human activity recognition, biomedical imaging etc.

Main objective

- Track the object(s) of interest as they move around during a video
- The state contains the current location, and typically information about scale, appearance and dynamics
- Not all trackers are equal : computational cost, handling occlusions, handling complex dynamics, handling the presence of visually similar targets, handling multiple target interactions

Filtering application : object tracking



A very popular and useful application in computer vision, with significant interest in human-robot interaction and autonomous systems, surveillance and traffic monitoring, human activity recognition, biomedical imaging etc.

Main objective

- Track the object(s) of interest as they move around during a video
- The state contains the current location, and typically information about scale, appearance and dynamics
- Not all trackers are equal : computational cost, handling occlusions, handling complex dynamics, handling the presence of visually similar targets, handling multiple target interactions

Filtering application : object tracking

Why not use the Kalman filter ?

Multiple problems

- A vanilla Kalman filter fails against almost all challenges from the previous slide ...
- Especially the simple transition model is very brittle
- Still, used extensively in the right context : constrained trajectories, high fps camera rate, e.g., [Bera et al., 2014]

Filtering application : object tracking

Why not use the Kalman filter ?

Multiple problems

- A vanilla Kalman filter fails against almost all challenges from the previous slide ...
- Especially the simple transition model is very brittle
- Still, used extensively in the right context : constrained trajectories, high fps camera rate, e.g., [Bera et al., 2014]

Filtering application : object tracking

Why not use the Kalman filter ?

Multiple problems

- A vanilla Kalman filter fails against almost all challenges from the previous slide ...
- Especially the simple transition model is very brittle
- Still, used extensively in the right context : constrained trajectories, high fps camera rate, e.g., [Bera et al., 2014]

Fig. 1: *Comparing AdaPT (bottom) with the Mean shift*

What else exists for tracking, beside filtering?

What is a good tracking alternative?

Data association

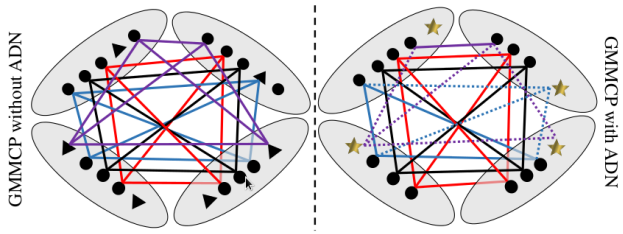
- A central task in tracking, based on evaluating a hypothesis on the current state, like an universal likelihood
- Solve medium-sized NP-hard problems efficiently
- Identify approximate formulations which scale to larger sizes

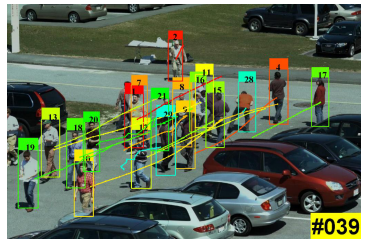
What else exists for tracking, beside filtering?

What is a good tracking alternative?

Data association

- A central task in tracking, based on evaluating a hypothesis on the current state, like an universal likelihood
- Solve medium-sized NP-hard problems efficiently
- Identify approximate formulations which scale to larger sizes





- Use a k-partite graph for data association (track = clique)
- Binary Integer Programming (NP-hard) and variations
- Very costly, but tractable up to batches of 50 frames and around 20 pedestrians

Dehghan, Afshin et al. "GMMCP Tracker: Globally Optimal Generalized Maximum Multi Clique Problem for Multiple Object Tracking." CVPR, 2015 [Dehghan et al., 2015a]

What else exists for tracking, beside filtering?

What is a good tracking alternative?

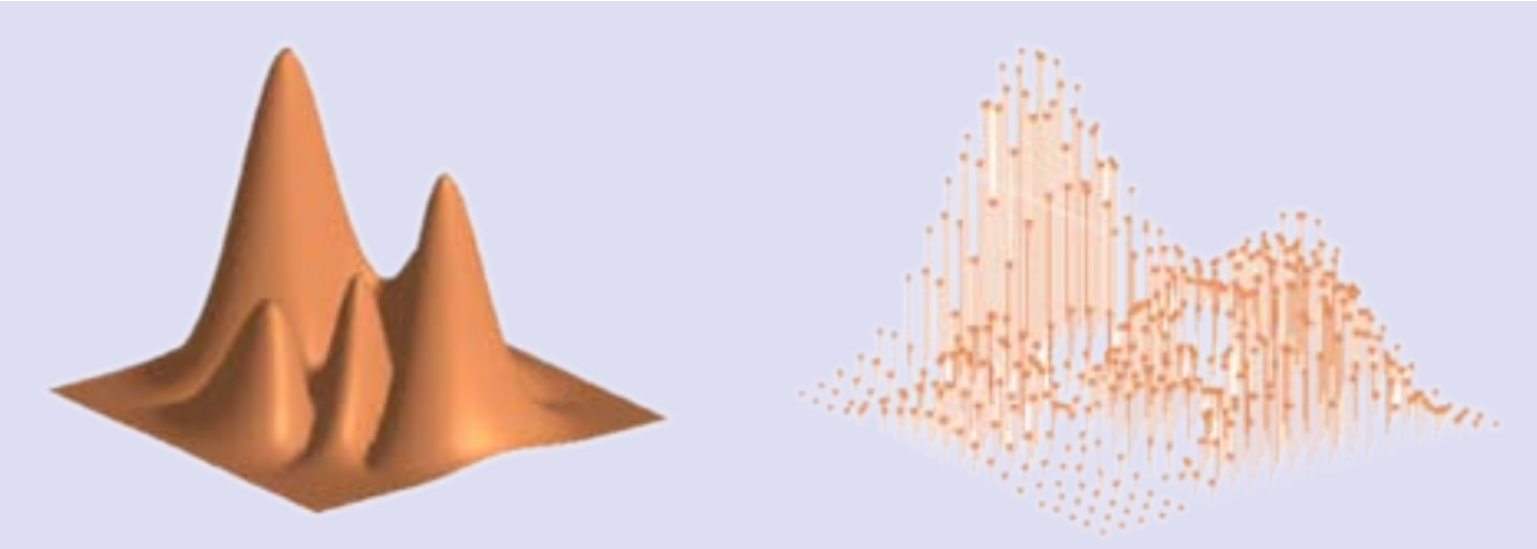
Data association

- A central task in tracking, based on evaluating a hypothesis on the current state, like an universal likelihood
- Solve medium-sized NP-hard problems efficiently
- Identify approximate formulations which scale to larger sizes

- Relax an IP problem to a LP or even to simpler forms
- Suited for enforcing consistency by integrating spatial constraints
- Also limited in practice to a few tens of targets - for the moment
- Suboptimal solutions are provided, but still a promising direction for coping with large number of targets

Chari, Visesh, et al. "On Pairwise Costs for Network Flow Multi-Object Tracking." CVPR, 2015 [Chari et al., 2015]
Dehghan, Afshin, et al. "Target identity-aware network flow for online multiple target tracking." CVPR, 2015 [Dehghan et al., 2015b]

Particle filtering



Principle : approach the posterior as a weighted sum of Dirac distributions corresponding to the filter particles

Monte Carlo approximation + Bayesian estimation

$$p(x_{0:t}|z_{1:t}) \approx \frac{1}{N} \sum_{i=1}^{N} w_t^{(i)} \delta_{x_{0:t}^{(i)}}(x_{0:t})$$

Particle filtering

Two founding papers

- "Novel approach to nonlinear/non-Gaussian Bayesian state estimation" [Gordon et al., 1993]
- "Condensation - conditional density propagation for visual tracking" [Isard and Blake, 1998]

Three main steps

1. Prediction step : estimate the prior density $p(x_{0:t}|z_{1:t-1})$ based on the transition $p(x_t|x_{t-1})$ and on the previous posterior $p(x_{0:t-1}|z_{1:t-1})$
2. Correction : $p(x_{0:t}|z_{1:t})$ based on the likelihood $p(y_t|x_t)$ and on the prior
3. Resampling step (selection)

Particle filtering

Two founding papers

- "Novel approach to nonlinear/non-Gaussian Bayesian state estimation" [Gordon et al., 1993]
- "Condensation - conditional density propagation for visual tracking" [Isard and Blake, 1998]

Three main steps

1. Prediction step : estimate the prior density $p(x_{0:t}|z_{1:t-1})$ based on the transition $p(x_t|x_{t-1})$ and on the previous posterior $p(x_{0:t-1}|z_{1:t-1})$
2. Correction : $p(x_{0:t}|z_{1:t})$ based on the likelihood $p(y_t|x_t)$ and on the prior
3. Resampling step (selection)

Particle filtering

Two founding papers

- "Novel approach to nonlinear/non-Gaussian Bayesian state estimation" [Gordon et al., 1993]
- "Condensation - conditional density propagation for visual tracking" [Isard and Blake, 1998]

Three main steps

1. Prediction step : estimate the prior density $p(x_{0:t}|z_{1:t-1})$ based on the transition $p(x_t|x_{t-1})$ and on the previous posterior $p(x_{0:t-1}|z_{1:t-1})$
2. Correction : $p(x_{0:t}|z_{1:t})$ based on the likelihood $p(y_t|x_t)$ and on the prior
3. Resampling step (selection)

Particle filtering

Two founding papers

- "Novel approach to nonlinear/non-Gaussian Bayesian state estimation" [Gordon et al., 1993]
- "Condensation - conditional density propagation for visual tracking" [Isard and Blake, 1998]

Three main steps

1. Prediction step : estimate the prior density $p(x_{0:t}|z_{1:t-1})$ based on the transition $p(x_t|x_{t-1})$ and on the previous posterior $p(x_{0:t-1}|z_{1:t-1})$
2. Correction : $p(x_{0:t}|z_{1:t})$ based on the likelihood $p(y_t|x_t)$ and on the prior
3. Resampling step (selection)

Particle filtering

A typical example in visual tracking

- the state and the state space : $x_t = \{x, y\} \in \mathbb{N}^2$
- a sample : $\{x_t^{(}i)\}_{i=1}^{N}$
- the transition function : $x_t^{(i)} \sim \mathcal{N}(\mu, \Sigma)$
- the likelihood function : $p(y_t|x_t) \propto e^{-\lambda d^2}$

Particle filtering

A typical example in visual tracking

- the state and the state space : $x_t = \{x, y\} \in \mathbb{N}^2$
- a sample : $\{x_t^{(}i)\}_{i=1}^{N}$
- the transition function : $x_t^{(i)} \sim \mathcal{N}(\mu, \Sigma)$
- the likelihood function : $p(y_t|x_t) \propto e^{-\lambda d^2}$

Particle filtering

A typical example in visual tracking

- the state and the state space : $x_t = \{x, y\} \in \mathbb{N}^2$
- a sample : $\{x_t^{(}i)\}_{i=1}^{N}$
- the transition function : $x_t^{(i)} \sim \mathcal{N}(\mu, \Sigma)$
- the likelihood function : $p(y_t|x_t) \propto e^{-\lambda d^2}$

Particle filtering

A typical example in visual tracking

- the state and the state space : $x_t = \{x, y\} \in \mathbb{N}^2$
- a sample : $\{x_t^{(}i)\}_{i=1}^{N}$
- the transition function : $x_t^{(i)} \sim \mathcal{N}(\mu, \Sigma)$
- the likelihood function : $p(y_t|x_t) \propto e^{-\lambda d^2}$

Importance sampling

Objective : approach the integral via realisations of a random variable distributed following the posterior distribution. Problem : we don't know how to sample along this distribution

Solution

Use a proposal function $q(x_{0:t}|z_{1:t})$ along which we know how to sample :

Importance sampling

Objective : approach the integral via realisations of a random variable distributed following the posterior distribution. Problem : we don't know how to sample along this distribution

Solution

Use a proposal function $q(x_{0:t}|z_{1:t})$ along which we know how to sample :

Importance sampling

Objective : approach the integral via realisations of a random variable distributed following the posterior distribution. Problem : we don't know how to sample along this distribution

Solution

Use a proposal function $q(x_{0:t}|z_{1:t})$ along which we know how to sample :

$$\begin{aligned}\mathbb{E}_p[\mathcal{F}(x_{0:t})] &= \int_{x_{0:t}} \mathcal{F}(x_{0:t}) \frac{p(x_{0:t}|z_{1:t})}{q(x_{0:t}|z_{1:t})} q(x_{0:t}|z_{1:t}) dx_{0:t} \\ &= \mathbb{E}_q \left[\mathcal{F}(x_{0:t}) \frac{p(x_{0:t}|z_{1:t})}{q(x_{0:t}|z_{1:t})} \right]\end{aligned}$$

Importance sampling

Solution

- N realisations $x_{0:t}^{(i)} \sim q(x_{0:t}|z_{1:t})$ which result in :

$$\hat{\mathbb{E}}_p[\mathcal{F}(x_{0:t})] = \frac{1}{N}\sum_{i=1}^{N} \mathcal{F}(x_{0:t}^{(i)}) \frac{p(x_{0:t}^{(i)}|z_{1:t})}{q(x_{0:t}^{(i)}|z_{1:t})}$$

- Thus the importance weighting :

$$w_t^{*(i)} = \frac{p(x_{0:t}^{(i)}|z_{1:t})}{q(x_{0:t}^{(i)}|z_{1:t})} = \frac{p(z_{1:t}|x_{0:t}^{(i)})p(x_{0:t}^{(i)})}{p(z_{1:t})q(x_{0:t}^{(i)}|z_{1:t})}$$

- Problem : $p(z_{1:t})$ is intractable

Importance sampling

Solution

- N realisations $x_{0:t}^{(i)} \sim q(x_{0:t}|z_{1:t})$ which result in :

$$\hat{\mathbb{E}}_p[\mathcal{F}(x_{0:t})] = \frac{1}{N}\sum_{i=1}^{N} \mathcal{F}(x_{0:t}^{(i)}) \frac{p(x_{0:t}^{(i)}|z_{1:t})}{q(x_{0:t}^{(i)}|z_{1:t})}$$

- Thus the importance weighting :

$$w_t^{*(i)} = \frac{p(x_{0:t}^{(i)}|z_{1:t})}{q(x_{0:t}^{(i)}|z_{1:t})} = \frac{p(z_{1:t}|x_{0:t}^{(i)})p(x_{0:t}^{(i)})}{p(z_{1:t})q(x_{0:t}^{(i)}|z_{1:t})}$$

- Problem : $p(z_{1:t})$ is intractable

Importance sampling

Solution

- N realisations $x_{0:t}^{(i)} \sim q(x_{0:t}|z_{1:t})$ which result in :

$$\hat{\mathbb{E}}_p[\mathcal{F}(x_{0:t})] = \frac{1}{N}\sum_{i=1}^{N} \mathcal{F}(x_{0:t}^{(i)}) \frac{p(x_{0:t}^{(i)}|z_{1:t})}{q(x_{0:t}^{(i)}|z_{1:t})}$$

- Thus the importance weighting :

$$w_t^{*(i)} = \frac{p(x_{0:t}^{(i)}|z_{1:t})}{q(x_{0:t}^{(i)}|z_{1:t})} = \frac{p(z_{1:t}|x_{0:t}^{(i)})p(x_{0:t}^{(i)})}{p(z_{1:t})q(x_{0:t}^{(i)}|z_{1:t})}$$

- Problem : $p(z_{1:t})$ is intractable

Importance sampling

Solution

- New estimator

$$\hat{\mathbb{E}}_p[\mathcal{F}(\mathrm{x}_{0:t})] = \frac{1}{N}\sum_{i=1}^{N}\mathcal{F}(\mathrm{x}_{0:t}^{(i)})\frac{w_t^{(i)}}{\sum_{j=1}^{N} w_t^{(j)}}$$

with $w_t^{(i)} \propto \frac{p(\mathrm{z}_{1:t}|\mathrm{x}_{0:t}^{(i)})p(\mathrm{x}_{0:t}^{(i)})}{q(\mathrm{x}_{0:t}^{(i)}|\mathrm{z}_{1:t})}$ or furthermore $w_t^{(i)} \propto \frac{p(\mathrm{x}_{0:t}^{(i)}|\mathrm{z}_{1:t})}{q(\mathrm{x}_{0:t}^{(i)}|\mathrm{z}_{1:t})}$

- Following the law of large numbers $\hat{\mathbb{E}}_p[\mathcal{F}(\mathrm{x}_{0:t})] \Rightarrow \mathbb{E}_p[\mathcal{F}(\mathrm{x}_{0:t})]$

Importance sampling

Solution

- New estimator

$$\hat{\mathbb{E}}_p[\mathcal{F}(\mathrm{x}_{0:t})] = \frac{1}{N}\sum_{i=1}^{N}\mathcal{F}(\mathrm{x}_{0:t}^{(i)})\frac{w_t^{(i)}}{\sum_{j=1}^{N} w_t^{(j)}}$$

  with $w_t^{(i)} \propto \frac{p(\mathrm{z}_{1:t}|\mathrm{x}_{0:t}^{(i)})p(\mathrm{x}_{0:t}^{(i)})}{q(\mathrm{x}_{0:t}^{(i)}|\mathrm{z}_{1:t})}$ or furthermore $w_t^{(i)} \propto \frac{p(\mathrm{x}_{0:t}^{(i)}|\mathrm{z}_{1:t})}{q(\mathrm{x}_{0:t}^{(i)}|\mathrm{z}_{1:t})}$

- Following the law of large numbers $\hat{\mathbb{E}}_p[\mathcal{F}(\mathrm{x}_{0:t})] \Rightarrow \mathbb{E}_p[\mathcal{F}(\mathrm{x}_{0:t})]$

Importance sampling

The recursive weight computation

- Hypothesis: $q(x_{0:t}^{(i)}|z_{1:t}) = q(x_t^{(i)}|x_{0:t-1}^{(i)}, z_{1:t})q(x_{0:t-1}^{(i)}|z_{1:t-1})$
- Then:

$$w_t^{(i)} \propto \frac{p(x_{0:t}^{(i)}|z_{1:t})}{q(x_{0:t}^{(i)}|z_{1:t})}$$

Importance sampling

The recursive weight computation

- Hypothesis: $q(x_{0:t}^{(i)}|z_{1:t}) = q(x_t^{(i)}|x_{0:t-1}^{(i)}, z_{1:t})q(x_{0:t-1}^{(i)}|z_{1:t-1})$
- Then:

$$w_t^{(i)} \propto \frac{p(x_{0:t}^{(i)}|z_{1:t})}{q(x_{0:t}^{(i)}|z_{1:t})}$$

Importance sampling

The recursive weight computation

- Hypothesis: $q(\mathrm{x}_{0:t}^{(i)}|\mathrm{z}_{1:t}) = q(\mathrm{x}_t^{(i)}|\mathrm{x}_{0:t-1}^{(i)}, \mathrm{z}_{1:t}) q(\mathrm{x}_{0:t-1}^{(i)}|\mathrm{z}_{1:t-1})$
- Then:

$$
\begin{aligned}
w_t^{(i)} &\propto \frac{p(\mathrm{x}_{0:t}^{(i)}|\mathrm{z}_{1:t})}{q(\mathrm{x}_{0:t}^{(i)}|\mathrm{z}_{1:t})} \\
&\propto \frac{p(\mathrm{z}_t|\mathrm{x}_t^{(i)}) p(\mathrm{x}_t^{(i)}|\mathrm{x}_{t-1}^{(i)}) p(\mathrm{x}_{0:t-1}^{(i)}|\mathrm{z}_{1:t-1})}{q(\mathrm{x}_t^{(i)}|\mathrm{x}_{0:t-1}^{(i)}, \mathrm{z}_{1:t}) q(\mathrm{x}_{0:t-1}^{(i)}|\mathrm{z}_{1:t-1})}
\end{aligned}
$$

Importance sampling

The recursive weight computation

- Hypothesis: $q(x_{0:t}^{(i)}|z_{1:t}) = q(x_t^{(i)}|x_{0:t-1}^{(i)}, z_{1:t}) q(x_{0:t-1}^{(i)}|z_{1:t-1})$
- Then:

$$
\begin{aligned}
w_t^{(i)} &\propto \frac{p(x_{0:t}^{(i)}|z_{1:t})}{q(x_{0:t}^{(i)}|z_{1:t})} \\
&\propto \frac{p(z_t|x_t^{(i)}) p(x_t^{(i)}|x_{t-1}^{(i)}) p(x_{0:t-1}^{(i)}|z_{1:t-1})}{q(x_t^{(i)}|x_{0:t-1}^{(i)}, z_{1:t}) q(x_{0:t-1}^{(i)}|z_{1:t-1})} \\
&\propto w_{t-1}^{(i)} \frac{p(z_t|x_t^{(i)}) p(x_t^{(i)}|x_{t-1}^{(i)})}{q(x_t^{(i)}|x_{0:t-1}^{(i)}, z_{1:t})}
\end{aligned}
$$

General algorithm

1. $p(x_{0:t-1}|z_{1:t-1})$ represented by $\{x_{0:t-1}^{(i)}, w_{t-1}^{(i)}\}_{i=1}^{N}$
2. Propagation via a proposal function :

$$x_t^{(i)} \sim q(x_t|x_{0:t-1}^{(i)}, z_{1:t})$$

3. Correction, or particle quality evaluation using the observations :

$$w_t^{(i)} \sim w_{t-1}^{(i)} \frac{p(z_t|x_t^{(i)})p(x_t^{(i)}|x_{t-1}^{(i)})}{q(x_t^{(i)}|x_{0:t-1}^{(i)}, z_{1:t})}, \tilde{w}_t^{(i)} = \frac{w_t^{(i)}}{\sum_{j=1}^{N} w_t^{(j)}}$$

4. Empirical posterior law at time t :

$$\mathbb{E}[\mathcal{F}(x_{0:t})] \approx \frac{1}{N} \sum_{i=1}^{N} \tilde{w}_t^{(i)} \mathcal{F}(x_{0:t}^{(i)})$$

5. Resampling (if necessary)

General algorithm

1. $p(x_{0:t-1}|z_{1:t-1})$ represented by $\{x_{0:t-1}^{(i)}, w_{t-1}^{(i)}\}_{i=1}^{N}$
2. Propagation via a proposal function :

$$x_t^{(i)} \sim q(x_t|x_{0:t-1}^{(i)}, z_{1:t})$$

3. Correction, or particle quality evaluation using the observations :

$$w_t^{(i)} \sim w_{t-1}^{(i)} \frac{p(z_t|x_t^{(i)})p(x_t^{(i)}|x_{t-1}^{(i)})}{q(x_t^{(i)}|x_{0:t-1}^{(i)}, z_{1:t})}, \tilde{w}_t^{(i)} = \frac{w_t^{(i)}}{\sum_{j=1}^{N} w_t^{(j)}}$$

4. Empirical posterior law at time t :

$$\mathbb{E}[\mathcal{F}(x_{0:t})] \approx \frac{1}{N} \sum_{i=1}^{N} \tilde{w}_t^{(i)} \mathcal{F}(x_{0:t}^{(i)})$$

5. Resampling (if necessary)

General algorithm

1. $p(x_{0:t-1}|z_{1:t-1})$ represented by $\{x_{0:t-1}^{(i)}, w_{t-1}^{(i)}\}_{i=1}^{N}$
2. Propagation via a proposal function :

$$x_t^{(i)} \sim q(x_t|x_{0:t-1}^{(i)}, z_{1:t})$$

3. Correction, or particle quality evaluation using the observations :

$$w_t^{(i)} \sim w_{t-1}^{(i)} \frac{p(z_t|x_t^{(i)})p(x_t^{(i)}|x_{t-1}^{(i)})}{q(x_t^{(i)}|x_{0:t-1}^{(i)}, z_{1:t})}, \tilde{w}_t^{(i)} = \frac{w_t^{(i)}}{\sum_{j=1}^{N} w_t^{(j)}}$$

4. Empirical posterior law at time t :

$$\mathbb{E}[\mathcal{F}(x_{0:t})] \approx \frac{1}{N} \sum_{i=1}^{N} \tilde{w}_t^{(i)} \mathcal{F}(x_{0:t}^{(i)})$$

5. Resampling (if necessary)

General algorithm

1. $p(x_{0:t-1}|z_{1:t-1})$ represented by $\{x_{0:t-1}^{(i)}, w_{t-1}^{(i)}\}_{i=1}^{N}$
2. Propagation via a proposal function :

$$x_t^{(i)} \sim q(x_t|x_{0:t-1}^{(i)}, z_{1:t})$$

3. Correction, or particle quality evaluation using the observations :

$$w_t^{(i)} \sim w_{t-1}^{(i)} \frac{p(z_t|x_t^{(i)})p(x_t^{(i)}|x_{t-1}^{(i)})}{q(x_t^{(i)}|x_{0:t-1}^{(i)}, z_{1:t})}, \tilde{w}_t^{(i)} = \frac{w_t^{(i)}}{\sum_{j=1}^{N} w_t^{(j)}}$$

4. Empirical posterior law at time t :

$$\mathbb{E}[\mathcal{F}(x_{0:t})] \approx \frac{1}{N} \sum_{i=1}^{N} \tilde{w}_t^{(i)} \mathcal{F}(x_{0:t}^{(i)})$$

5. Resampling (if necessary)

General algorithm

1. $p(x_{0:t-1}|z_{1:t-1})$ represented by $\{x_{0:t-1}^{(i)}, w_{t-1}^{(i)}\}_{i=1}^{N}$
2. Propagation via a proposal function :

$$x_t^{(i)} \sim q(x_t|x_{0:t-1}^{(i)}, z_{1:t})$$

3. Correction, or particle quality evaluation using the observations :

$$w_t^{(i)} \sim w_{t-1}^{(i)} \frac{p(z_t|x_t^{(i)})p(x_t^{(i)}|x_{t-1}^{(i)})}{q(x_t^{(i)}|x_{0:t-1}^{(i)}, z_{1:t})}, \tilde{w}_t^{(i)} = \frac{w_t^{(i)}}{\sum_{j=1}^{N} w_t^{(j)}}$$

4. Empirical posterior law at time t :

$$\mathbb{E}[\mathcal{F}(x_{0:t})] \approx \frac{1}{N} \sum_{i=1}^{N} \tilde{w}_t^{(i)} \mathcal{F}(x_{0:t}^{(i)})$$

5. Resampling (if necessary)

The prediction function q

Choosing the prediction function

- The usual choice is : $q(x_t^{(i)}|x_{0:t-1}^{(i)}, z_{1:t}) = p(x_t^{(i)}|x_{t-1}^{(i)})$
- This means that :

$$w_t^{(i)} \sim w_{t-1}^{(i)} \frac{p(z_t|x_t^{(i)})p(x_t^{(i)}|x_{t-1}^{(i)})}{q(x_t^{(i)}|x_{0:t-1}^{(i)}, z_{1:t})} \sim w_{t-1}^{(i)} p(z_t|x_t^{(i)})$$

- The weight is proportional to the likelihood : the bootstrap filter
- In practice : $q \sim \mathcal{N}(x_{t-1}, \Sigma)$

The prediction function q

Choosing the prediction function

- The usual choice is : $q(\mathrm{x}_t^{(i)}|\mathrm{x}_{0:t-1}^{(i)}, \mathrm{z}_{1:t}) = p(\mathrm{x}_t^{(i)}|\mathrm{x}_{t-1}^{(i)})$
- This means that :

$$w_t^{(i)} \sim w_{t-1}^{(i)} \frac{p(\mathrm{z}_t|\mathrm{x}_t^{(i)})p(\mathrm{x}_t^{(i)}|\mathrm{x}_{t-1}^{(i)})}{q(\mathrm{x}_t^{(i)}|\mathrm{x}_{0:t-1}^{(i)}, \mathrm{z}_{1:t})} \sim w_{t-1}^{(i)} p(\mathrm{z}_t|\mathrm{x}_t^{(i)})$$

- The weight is proportional to the likelihood : the bootstrap filter
- In practice : $q \sim \mathcal{N}(\mathrm{x}_{t-1}, \Sigma)$

The prediction function q

Choosing the prediction function

- The usual choice is : $q(\mathrm{x}_t^{(i)}|\mathrm{x}_{0:t-1}^{(i)}, \mathrm{z}_{1:t}) = p(\mathrm{x}_t^{(i)}|\mathrm{x}_{t-1}^{(i)})$
- This means that :

$$w_t^{(i)} \sim w_{t-1}^{(i)} \frac{p(\mathrm{z}_t|\mathrm{x}_t^{(i)})p(\mathrm{x}_t^{(i)}|\mathrm{x}_{t-1}^{(i)})}{q(\mathrm{x}_t^{(i)}|\mathrm{x}_{0:t-1}^{(i)}, \mathrm{z}_{1:t})} \sim w_{t-1}^{(i)} p(\mathrm{z}_t|\mathrm{x}_t^{(i)})$$

- The weight is proportional to the likelihood : the bootstrap filter
- In practice : $q \sim \mathcal{N}(\mathrm{x}_{t-1}, \Sigma)$

The prediction function q

Choosing the prediction function

- The usual choice is : $q(\mathrm{x}_t^{(i)}|\mathrm{x}_{0:t-1}^{(i)}, \mathrm{z}_{1:t}) = p(\mathrm{x}_t^{(i)}|\mathrm{x}_{t-1}^{(i)})$
- This means that :

$$w_t^{(i)} \sim w_{t-1}^{(i)} \frac{p(\mathrm{z}_t|\mathrm{x}_t^{(i)})p(\mathrm{x}_t^{(i)}|\mathrm{x}_{t-1}^{(i)})}{q(\mathrm{x}_t^{(i)}|\mathrm{x}_{0:t-1}^{(i)}, \mathrm{z}_{1:t})} \sim w_{t-1}^{(i)} p(\mathrm{z}_t|\mathrm{x}_t^{(i)})$$

- The weight is proportional to the likelihood : the bootstrap filter
- In practice : $q \sim \mathcal{N}(\mathrm{x}_{t-1}, \Sigma)$

The correction step

Compute the posterior from the prior

- Based on the likelihood function $p(z_t|x_t)$
- The belief in the correctness of the observation z_t given the target's state x_t
- We need to evaluate the coherence between the model and the hypothesis
- How do we get the model ? based on the previous estimation
- How do we get the hypothesis ? it's a particle at time t
- In practice $p(y_t|x_t) \propto e^{-\lambda d^2}$

The correction step

Compute the posterior from the prior

- Based on the likelihood function $p(z_t|x_t)$
- The belief in the correctness of the observation z_t given the target's state x_t
- We need to evaluate the coherence between the model and the hypothesis
- How do we get the model ? based on the previous estimation
- How do we get the hypothesis ? it's a particle at time t
- In practice $p(y_t|x_t) \propto e^{-\lambda d^2}$

The correction step

Compute the posterior from the prior

- Based on the likelihood function $p(z_t|x_t)$
- The belief in the correctness of the observation z_t given the target's state x_t
- We need to evaluate the coherence between the model and the hypothesis
- How do we get the model ? based on the previous estimation
- How do we get the hypothesis ? it's a particle at time t
- In practice $p(y_t|x_t) \propto e^{-\lambda d^2}$

The correction step

Compute the posterior from the prior

- Based on the likelihood function $p(z_t|x_t)$
- The belief in the correctness of the observation z_t given the target's state x_t
- We need to evaluate the coherence between the model and the hypothesis
- How do we get the model ? based on the previous estimation
- How do we get the hypothesis ? it's a particle at time t
- In practice $p(y_t|x_t) \propto e^{-\lambda d^2}$

The correction step

Compute the posterior from the prior

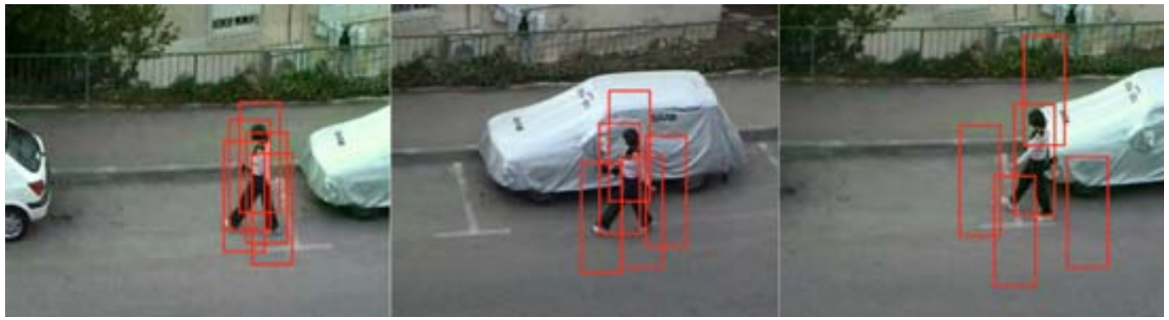
- Based on the likelihood function $p(z_t|x_t)$
- The belief in the correctness of the observation z_t given the target's state x_t
- We need to evaluate the coherence between the model and the hypothesis
- How do we get the model ? based on the previous estimation
- How do we get the hypothesis ? it's a particle at time t
- In practice $p(y_t|x_t) \propto e^{-\lambda d^2}$

The correction step

Compute the posterior from the prior

- Based on the likelihood function $p(z_t|x_t)$
- The belief in the correctness of the observation z_t given the target's state x_t
- We need to evaluate the coherence between the model and the hypothesis
- How do we get the model ? based on the previous estimation
- How do we get the hypothesis ? it's a particle at time t
- In practice $p(y_t|x_t) \propto e^{-\lambda d^2}$

The resampling step



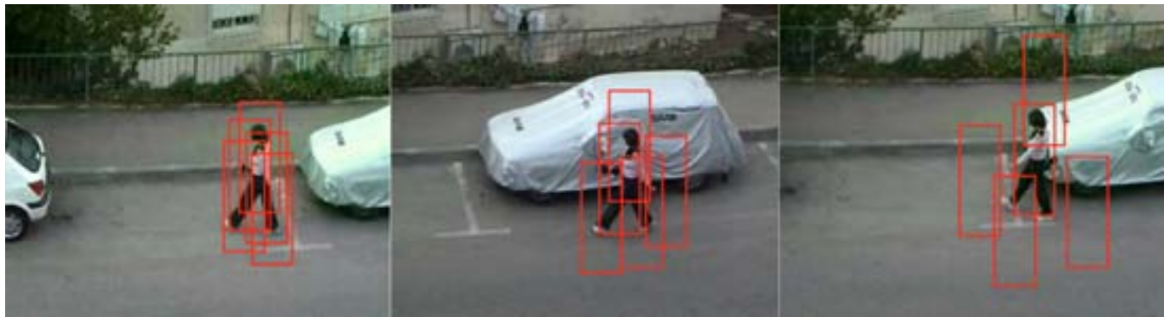
Sample degenerescence

- Unavoidable : particle drift $\Rightarrow$ filter divergence
- Solution : resampling - duplicating the particles with the largest weights
- In practice : multinomial sampling $k \in \mathcal{U}(0,1), i = 1, \ldots, N;$:

$$S(t) = \{x_t^{(i)}, w_t^{(i)}\} \Rightarrow S'(t) = \{x_t^{(D(k_i))}, 1/N\}$$

with $D(k_i)$ an integer such that $\sum_h^{D(k_i)-1} w_t^{(h)} < k_i < \sum_h^{D(k_i)} w_t^{(h)}$

The resampling step



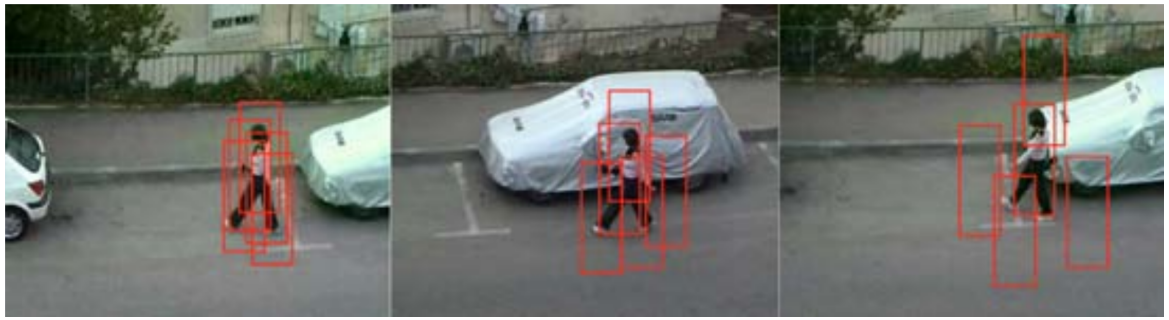
Sample degenerescence

- Unavoidable : particle drift $\Rightarrow$ filter divergence
- Solution : resampling - duplicating the particles with the largest weights
- In practice : multinomial sampling $k \in \mathcal{U}(0,1), i = 1, \ldots, N;$:

$$S(t) = \{x_t^{(i)}, w_t^{(i)}\} \Rightarrow S'(t) = \{x_t^{(D(k_i))}, 1/N\}$$

with $D(k_i)$ an integer such that $\sum_h^{D(k_i)-1} w_t^{(h)} < k_i < \sum_h^{D(k_i)} w_t^{(h)}$

The resampling step



Sample degenerescence

- Unavoidable : particle drift $\Rightarrow$ filter divergence
- Solution : resampling - duplicating the particles with the largest weights
- In practice : multinomial sampling $k \in \mathcal{U}(0,1), i = 1, \ldots, N;$:

$$S(t) = \{x_t^{(i)}, w_t^{(i)}\} \Rightarrow S'(t) = \{x_t^{(D(k_i))}, 1/N\}$$

with $D(k_i)$ an integer such that $\sum_h^{D(k_i)-1} w_t^{(h)} < k_i < \sum_h^{D(k_i)} w_t^{(h)}$

Outline

Filtering and learning

Particle filtering : not practical in the case of multiple targets (which may also interact).
How to use learning to

- decrease the cost at least at execution time
- also improve performance with respect to alternative solutions (hopefully)?

Crowd tracking

- Persistent occlusions
- Homogeneous targets
- Small or strongly varying scales accross the image space

Filtering and learning

Particle filtering : not practical in the case of multiple targets (which may also interact).
How to use learning to

- decrease the cost at least at execution time
- also improve performance with respect to alternative solutions (hopefully)?

Crowd tracking

- Persistent occlusions
- Homogeneous targets
- Small or strongly varying scales accross the image space

Filtering and learning

Particle filtering : not practical in the case of multiple targets (which may also interact).
How to use learning to

- decrease the cost at least at execution time
- also improve performance with respect to alternative solutions (hopefully)?

Crowd tracking

- Persistent occlusions
- Homogeneous targets
- Small or strongly varying scales accross the image space

Filtering and learning

Particle filtering : not practical in the case of multiple targets (which may also interact).
How to use learning to

- decrease the cost at least at execution time
- also improve performance with respect to alternative solutions (hopefully)?

Crowd tracking

- Persistent occlusions
- Homogeneous targets
- Small or strongly varying scales accross the image space

Filtering and learning

Particle filtering : not practical in the case of multiple targets (which may also interact).
How to use learning to

- decrease the cost at least at execution time
- also improve performance with respect to alternative solutions (hopefully)?

Crowd tracking

- Persistent occlusions
- Homogeneous targets
- Small or strongly varying scales accross the image space

Crowd state - dynamics and visual appearance

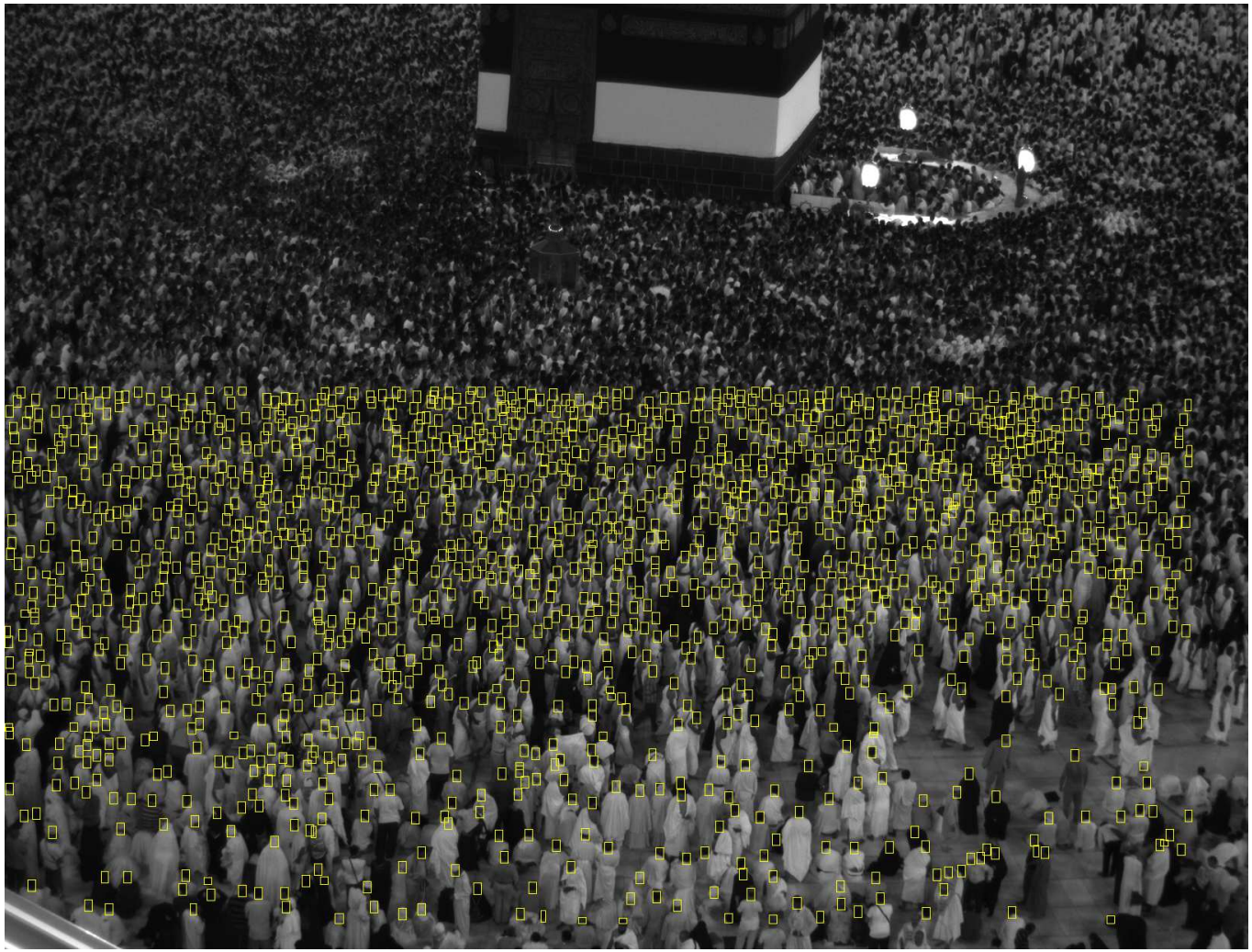
Crowd state - dynamics and visual appearance

Crowd state - dynamics and visual appearance

Crowd state - dynamics and visual appearance

Crowd state - dynamics and visual appearance

Crowd state - dynamics and visual appearance



Filtering and learning

How to make a tracker work on Makkah data? [Franchi et al., 2020a]

About the data

- Around 12 annotated images for training the detector
- A sequence of 50 images and about 620 tracks for evaluation

Algorithms involved

- Detector : RetinaNet [Lin et al., 2017]
- O. Flow : DeepFlow [Weinzaepfel et al., 2013]
- DCNN Corrector

Filtering and learning

How to make a tracker work on Makkah data? [Franchi et al., 2020a]

About the data

- Around 12 annotated images for training the detector
- A sequence of 50 images and about 620 tracks for evaluation

Algorithms involved

- Detector : RetinaNet [Lin et al., 2017]
- O. Flow : DeepFlow [Weinzaepfel et al., 2013]
- DCNN Corrector

Filtering and learning

How to make a tracker work on Makkah data? [Franchi et al., 2020a]

About the data

- Around 12 annotated images for training the detector
- A sequence of 50 images and about 620 tracks for evaluation

Algorithms involved

- Detector : RetinaNet [Lin et al., 2017]
- O. Flow : DeepFlow [Weinzaepfel et al., 2013]
- DCNN Corrector

Filtering and learning

How to make a tracker work on Makkah data? [Franchi et al., 2020a]

About the data

- Around 12 annotated images for training the detector
- A sequence of 50 images and about 620 tracks for evaluation

Algorithms involved

- Detector : RetinaNet [Lin et al., 2017]
- O. Flow : DeepFlow [Weinzaepfel et al., 2013]
- DCNN Corrector

Filtering and learning

How to make a tracker work on Makkah data? [Franchi et al., 2020a]

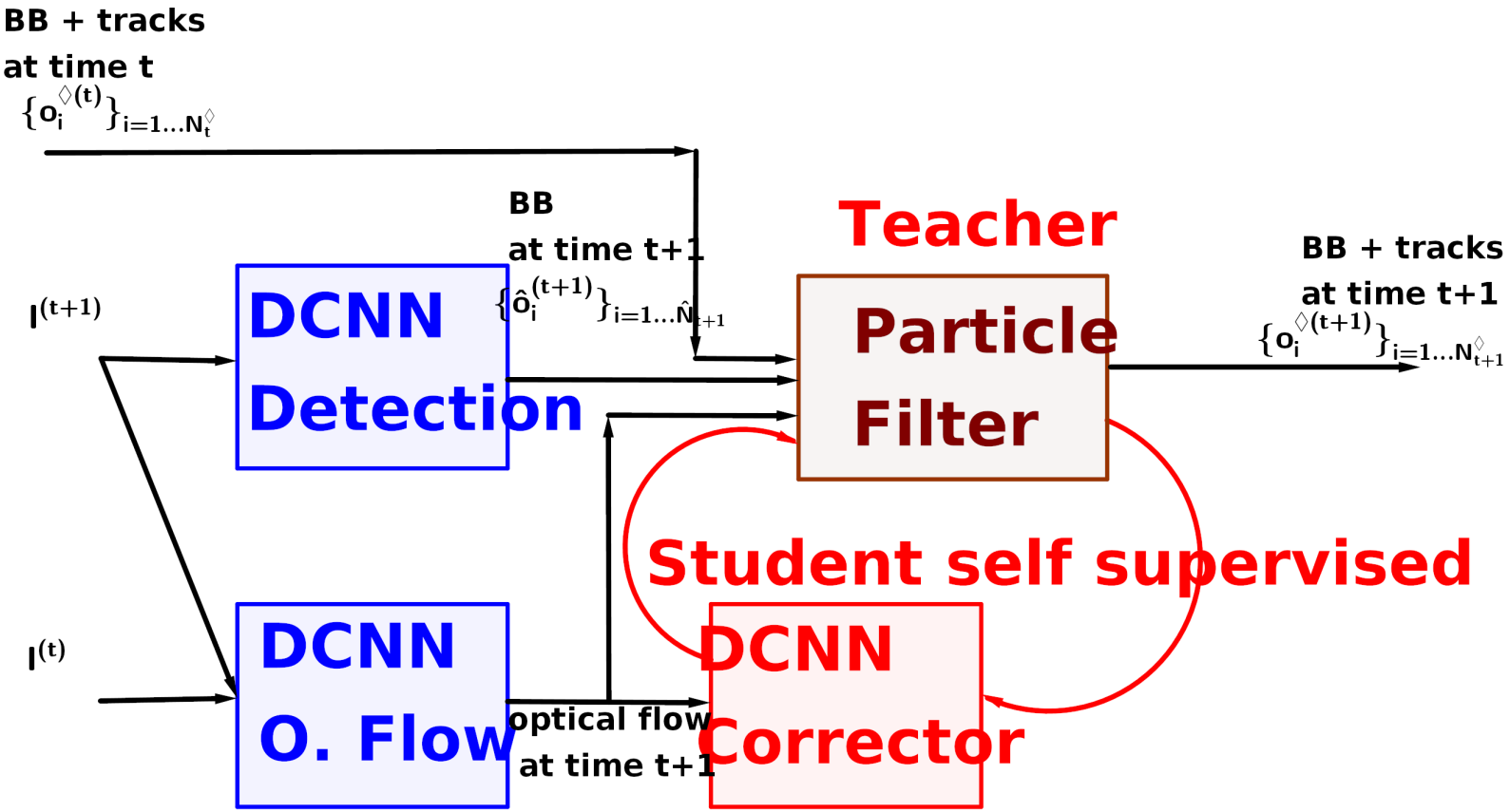
About the data

- Around 12 annotated images for training the detector
- A sequence of 50 images and about 620 tracks for evaluation

Algorithms involved

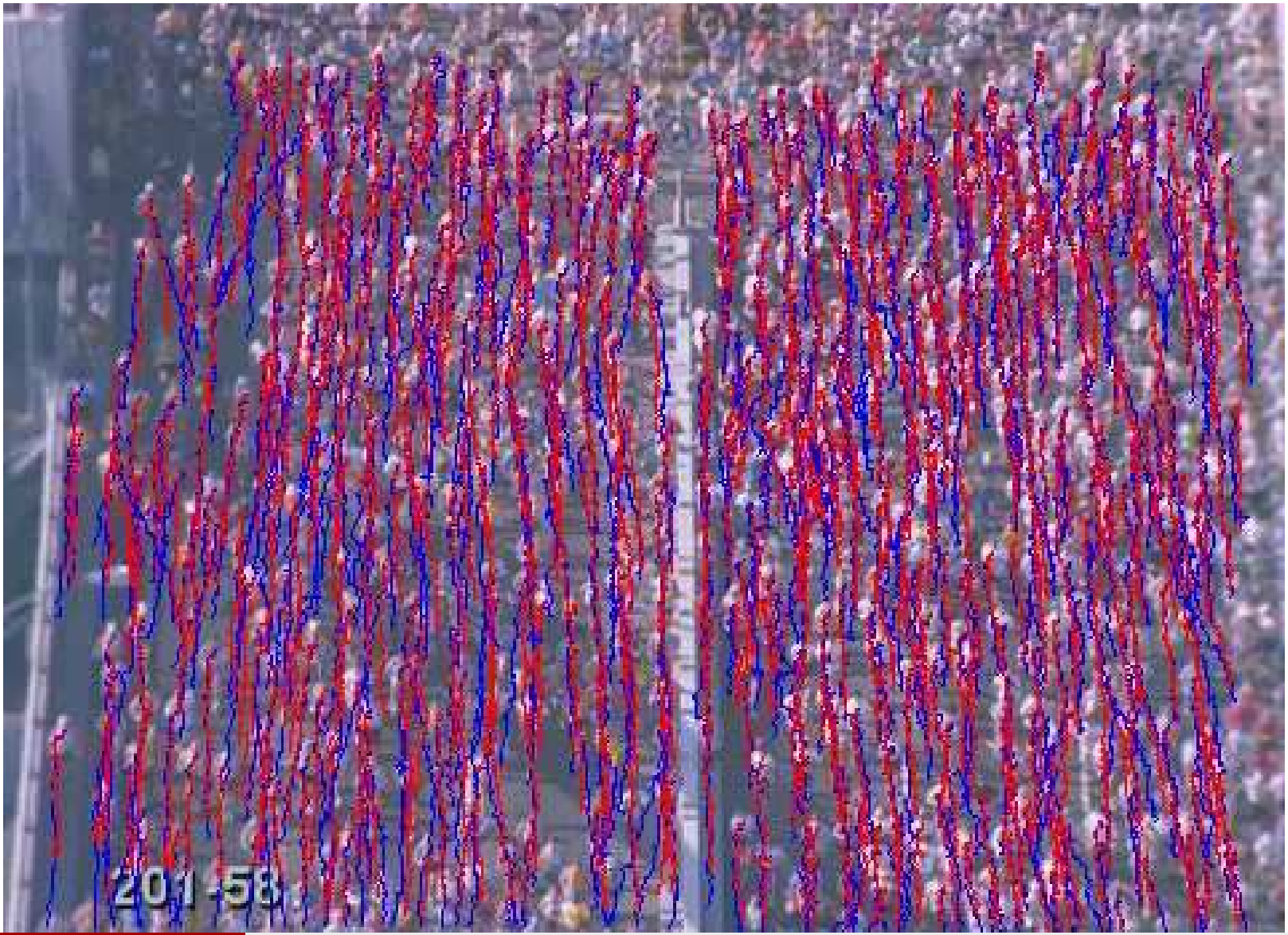
- Detector : RetinaNet [Lin et al., 2017]
- O. Flow : DeepFlow [Weinzaepfel et al., 2013]
- DCNN Corrector

Proposed approach



Proposed approach

Proposed approach



Proposed approach

	MOTP	MOTA	TA
flow	22%	16%	39%
NMC	54%	61%	78%
PF	59%	66%	73%
PFMRF	60%	70%	78%
PFMRF DCNN Cor 5 epochs $T_{\text{data}} = 5$	**64%**	**75%**	**84%**

Table: Performance of the tracking algorithms on the Makkah dataset.

Outline

Early and late fusion with deep learning

First, how do we define what early vs late fusion means?

- It is quite uncommon to perform actual early fusion on heterogeneous, large amounts of data : too costly
- Early fusion is most of the times some kind of intermediate fusion on low level modality representations
- Late fusion algorithms are more universal, but they require uncertainty estimation for the sources

Early and late fusion with deep learning

First, how do we define what early vs late fusion means?

- It is quite uncommon to perform actual early fusion on heterogeneous, large amounts of data : too costly
- Early fusion is most of the times some kind of intermediate fusion on low level modality representations
- Late fusion algorithms are more universal, but they require uncertainty estimation for the sources

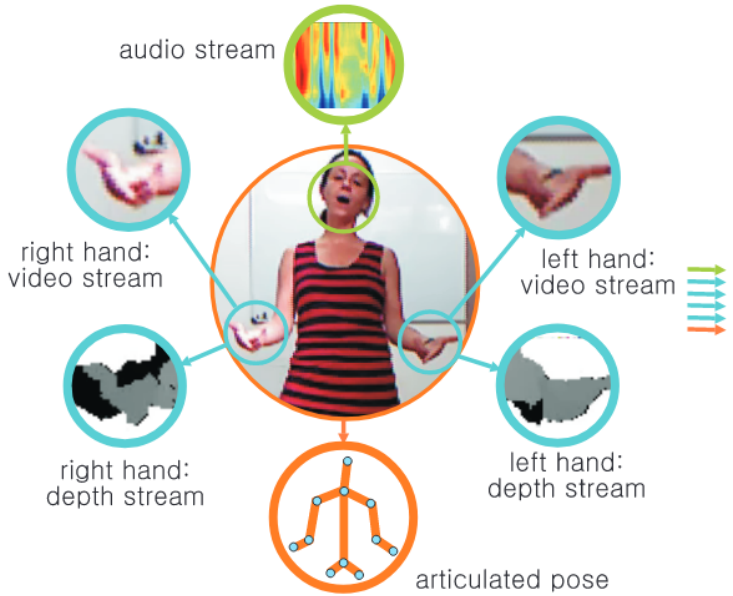
Early and late fusion with deep learning

First, how do we define what early vs late fusion means?

- It is quite uncommon to perform actual early fusion on heterogeneous, large amounts of data : too costly
- Early fusion is most of the times some kind of intermediate fusion on low level modality representations
- Late fusion algorithms are more universal, but they require uncertainty estimation for the sources

Why do we need to do early fusion with deep networks?

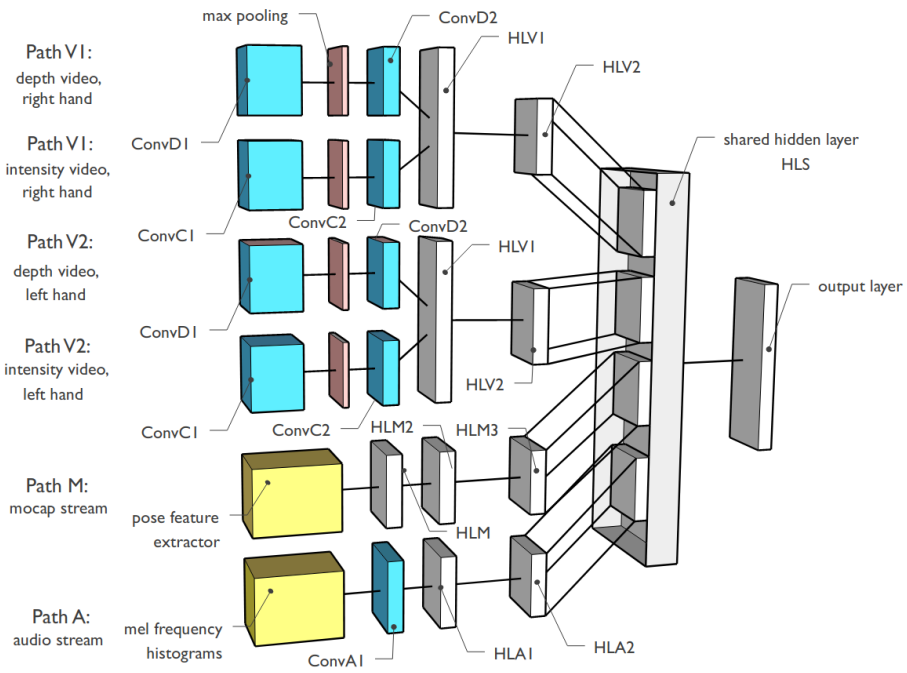
Fusing different modalities



- Can we improve the performance with early fusion? Answer : yes, but 1) it depends on the problem, 2) it is tricky and 3) it is quite costly [Neverova et al., 2016]

Fusion steps

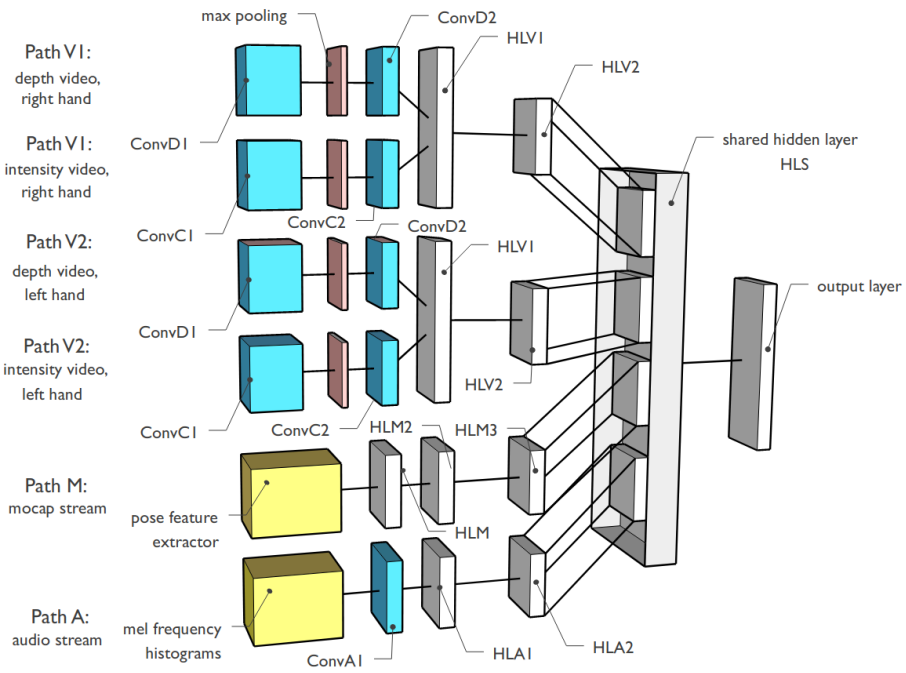
Fusing different modalities



- Pre-training must be performed independently for each modality
- Introduce a shared hidden layer
- Do not wire all the pre-trained paths to the shared layer (quick degradation of the pre-trained connections)
- Gradually add the new paths

Fusion steps

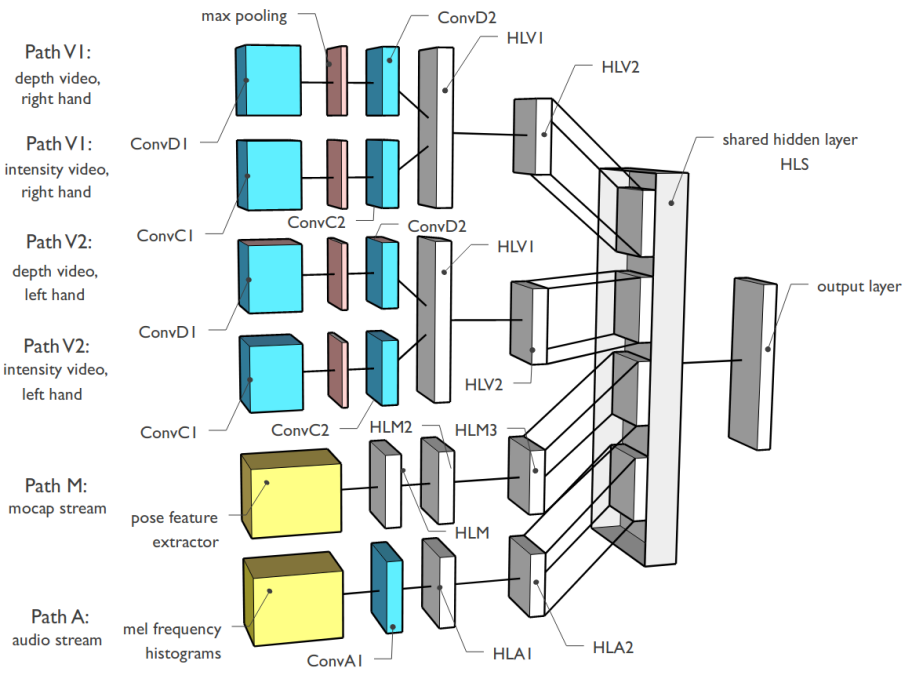
Fusing different modalities



- Pre-training must be performed independently for each modality
- Introduce a shared hidden layer
- Do not wire all the pre-trained paths to the shared layer (quick degradation of the pre-trained connections)
- Gradually add the new paths

Fusion steps

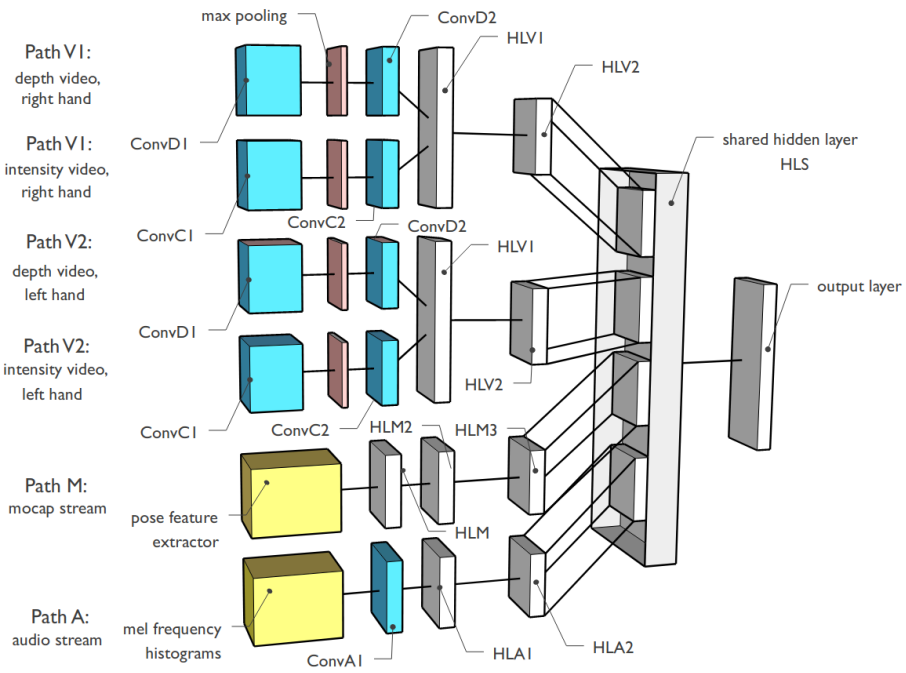
Fusing different modalities



- Pre-training must be performed independently for each modality
- Introduce a shared hidden layer
- Do not wire all the pre-trained paths to the shared layer (quick degradation of the pre-trained connections)
- Gradually add the new paths

Fusion steps

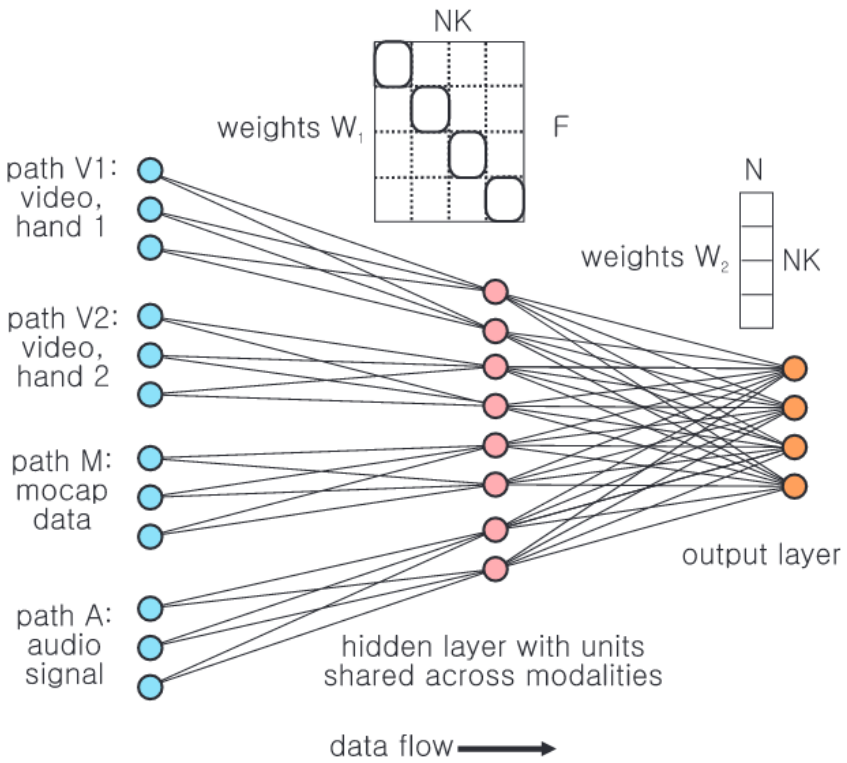
Fusing different modalities

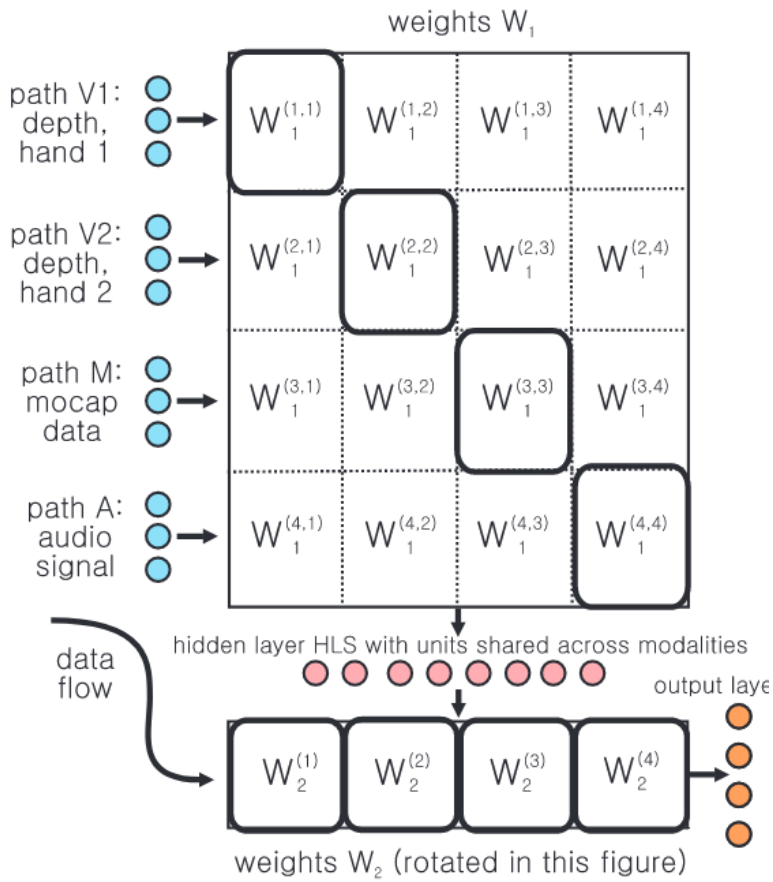


- Pre-training must be performed independently for each modality
- Introduce a shared hidden layer
- Do not wire all the pre-trained paths to the shared layer (quick degradation of the pre-trained connections)
- Gradually add the new paths

Fusion steps

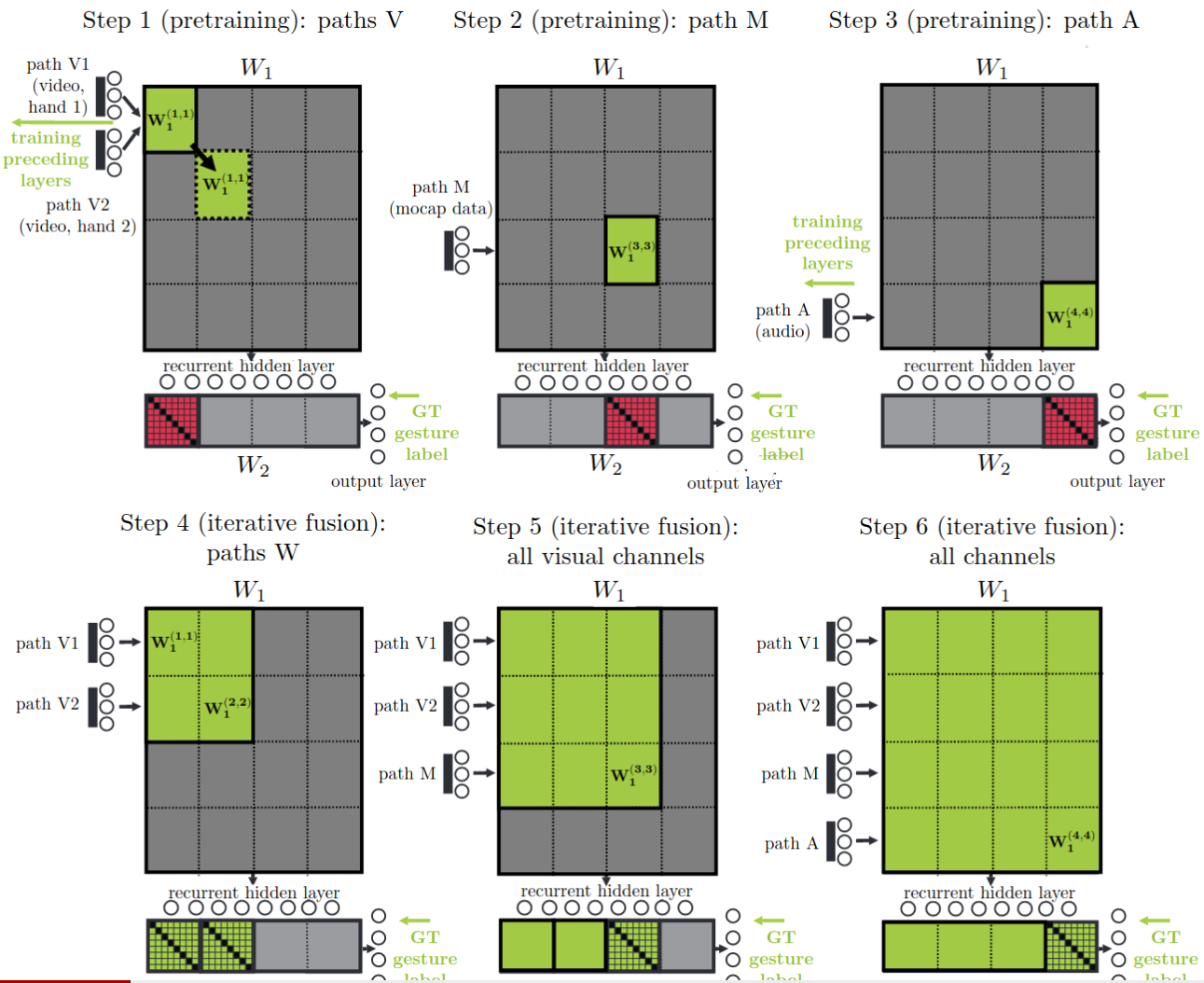
Fusing different modalities





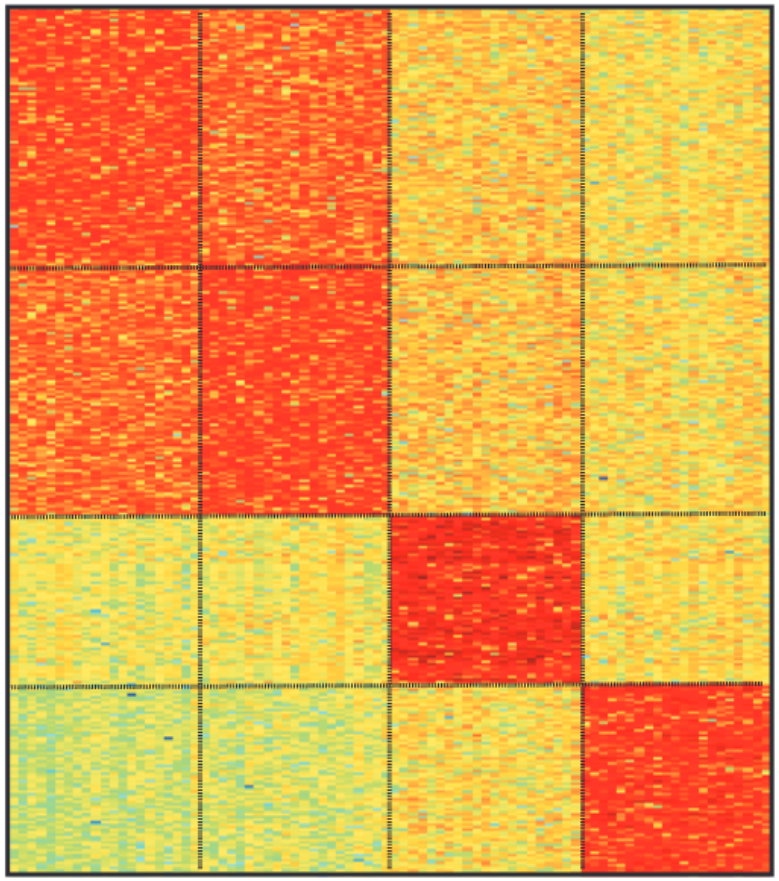
Fusion steps

Fusing different modalities



Fusion steps

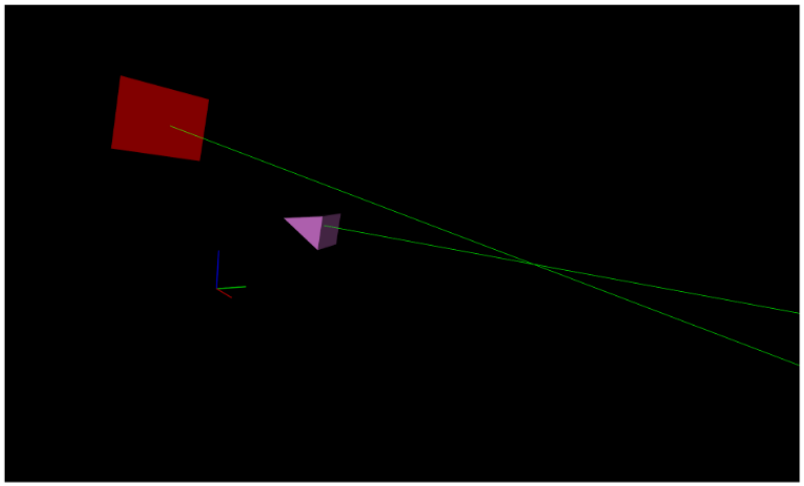
Fusing different modalities



Data association application

Pedestrian localization: a data association problem

(a) Top view (b) Egocentric view



Data association application

Pedestrian localization: a data association problem

- Scene geometry is supposed to be known
- Pedestrian detections may still be ambiguous
- Geometry estimation may be imprecise
- Data association : usually solved using the Hungarian algorithm:

$$\min \sum_{i=1}^{M} \sum_{j=1}^{N} c_{ij} x_{ij}, \text{ subject to}$$

$$\sum_{i=1}^{M} x_{ij} \leqslant 1, \forall j; \quad \sum_{j=1}^{N} x_{ij} \leqslant 1, \forall i; \quad x_{ij} \in \{0, 1\}, \forall i, j.$$

Data association application

Pedestrian localization: a data association problem

- Scene geometry is supposed to be known
- Pedestrian detections may still be ambiguous
- Geometry estimation may be imprecise
- Data association : usually solved using the Hungarian algorithm:

$$\min \sum_{i=1}^{M} \sum_{j=1}^{N} c_{ij} x_{ij}, \text{ subject to}$$

$$\sum_{i=1}^{M} x_{ij} \leqslant 1, \forall j; \quad \sum_{j=1}^{N} x_{ij} \leqslant 1, \forall i; \quad x_{ij} \in \{0, 1\}, \forall i, j.$$

Data association application

Pedestrian localization: a data association problem

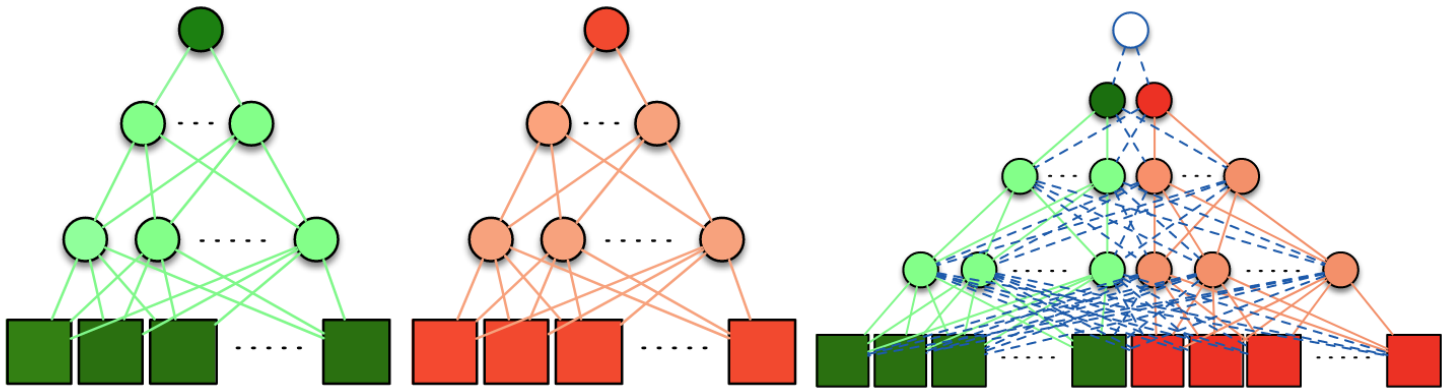
- Scene geometry is supposed to be known
- Pedestrian detections may still be ambiguous
- Geometry estimation may be imprecise
- Data association : usually solved using the Hungarian algorithm:

$$\min \sum_{i=1}^{M} \sum_{j=1}^{N} c_{ij} x_{ij}, \text{ subject to}$$

$$\sum_{i=1}^{M} x_{ij} \leqslant 1, \forall j; \quad \sum_{j=1}^{N} x_{ij} \leqslant 1, \forall i; \quad x_{ij} \in \{0, 1\}, \forall i, j.$$

Data association application

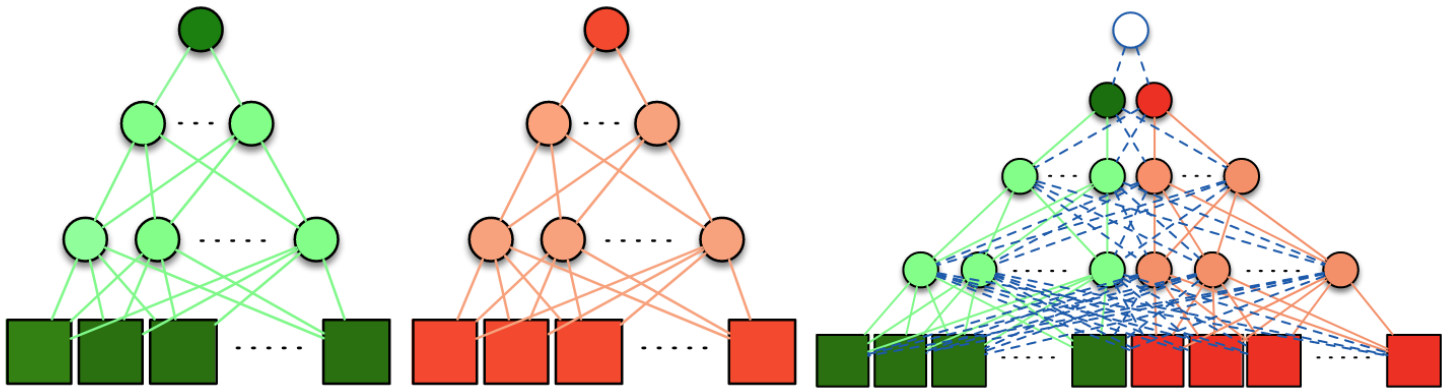
Pedestrian localization: a data association problem



(a) Visual MLP (b) Geometric MLP (c) Joint MLP

Data association application

Pedestrian localization: a data association problem



(a) Visual MLP (b) Geometric MLP (c) Joint MLP

- Cost for visual descriptors is straightforward
- Geometry parametrization should be invariant and numerically stable

Data association application

Cost \ (Det.;Desc.)	Cheirality	(YOLO;hist) Explicit	(YOLO;hist) Learned	(YOLO; desc1) Explicit	(YOLO; desc1) Learned	(Idemia;desc2) Explicit	(Idemia;desc2) Learned
Visual only	No	43.08	43.78	84.73	71.22	43.82	58.64
	Yes	43.11	43.98	84.73	71.22	43.81	58.72
Geometry only	No	44.26	87.24	44.26	87.24	45.01	86.06
	Yes	44.86	87.24	44.86	87.24	46.03	86.06
Sum	No	54.01	87.70	66.45	88.49	58.59	86.33
	Yes	54.60	87.70	66.81	**88.49**	59.19	86.33
Concatenation-free	No	-	87.70	-	88.42	-	86.74
	Yes	-	87.70	-	88.42	-	86.74
Concatenation-Freeze	No	-	87.67	-	88.06	-	86.71
	Yes	-	87.67	-	88.06	-	86.71
Concatenation-unfreeze	No	-	**87.74**	-	88.38	-	**86.83**
	Yes	-	**87.74**	-	88.38	-	**86.83**
Fusion-sup	-	64.84	89.55	88.36	90.55	64.84	88.55

Why do we need to do early fusion with deep networks?

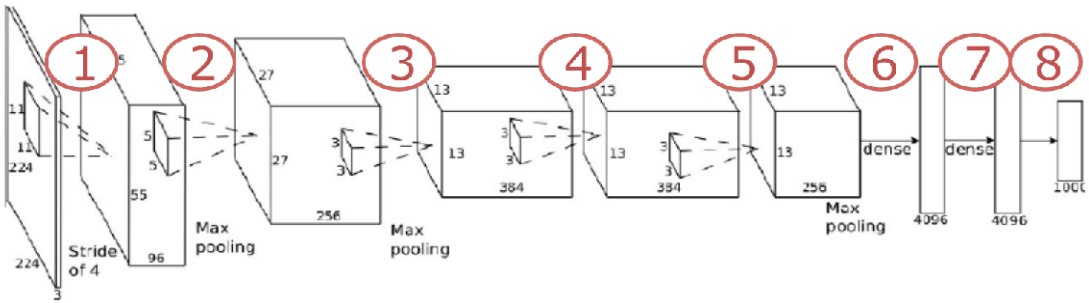
- deep models tend to have a lot of parameters $\Rightarrow$ great need for large, high-quality datasets for training
- the first part of the network : representation learning; final part : the decision process

Why do we need to do early fusion with deep networks?

- deep models tend to have a lot of parameters $\Rightarrow$ great need for large, high-quality datasets for training
- the first part of the network : representation learning; final part : the decision process

Why do we need to do early fusion with deep networks?

- deep models tend to have a lot of parameters ⇒ great need for large, high-quality datasets for training
- the first part of the network : representation learning; final part : the decision process



Why do we need to do early fusion with deep networks?

Relying on multiple datasets

Dataset evolution in classification

- 1998 : MNIST, 10 classes and 60k+10k images
- 2004 : Caltech 101, 100 classes and 9k images
- 2009 : CIFAR 10, 10 classes and 60k images
- 2009 : ImageNet, 1000 classes and 10M images
- for many recent datasets, training (but not running) a model is prohibitively costly.
- Solution : use off-the-shelf model, keep the representation extraction part, adapt the output layers for your own specific task $\Rightarrow$ finetuning
- One may want to finetune N models, then combine their decisions $\Rightarrow$ late fusion

Why do we need to do early fusion with deep networks?

Relying on multiple datasets

Dataset evolution in classification

- 1998 : MNIST, 10 classes and 60k+10k images
- 2004 : Caltech 101, 100 classes and 9k images
- 2009 : CIFAR 10, 10 classes and 60k images
- 2009 : ImageNet, 1000 classes and 10M images
- for many recent datasets, training (but not running) a model is prohibitively costly.
- Solution : use off-the-shelf model, keep the representation extraction part, adapt the output layers for your own specific task $\Rightarrow$ finetuning
- One may want to finetune N models, then combine their decisions $\Rightarrow$ late fusion

Why do we need to do early fusion with deep networks?

Relying on multiple datasets

Dataset evolution in classification

- 1998 : MNIST, 10 classes and 60k+10k images
- 2004 : Caltech 101, 100 classes and 9k images
- 2009 : CIFAR 10, 10 classes and 60k images
- 2009 : ImageNet, 1000 classes and 10M images
- for many recent datasets, training (but not running) a model is prohibitively costly.
- Solution : use off-the-shelf model, keep the representation extraction part, adapt the output layers for your own specific task $\Rightarrow$ finetuning
- One may want to finetune N models, then combine their decisions $\Rightarrow$ late fusion

Why do we need to do early fusion with deep networks?

Relying on multiple datasets

Dataset evolution in classification

- 1998 : MNIST, 10 classes and 60k+10k images
- 2004 : Caltech 101, 100 classes and 9k images
- 2009 : CIFAR 10, 10 classes and 60k images
- 2009 : ImageNet, 1000 classes and 10M images
- for many recent datasets, training (but not running) a model is prohibitively costly.
- Solution : use off-the-shelf model, keep the representation extraction part, adapt the output layers for your own specific task $\Rightarrow$ finetuning
- One may want to finetune N models, then combine their decisions $\Rightarrow$ late fusion

Why do we need to do early fusion with deep networks?

Relying on multiple datasets

Dataset evolution in classification

- 1998 : MNIST, 10 classes and 60k+10k images
- 2004 : Caltech 101, 100 classes and 9k images
- 2009 : CIFAR 10, 10 classes and 60k images
- 2009 : ImageNet, 1000 classes and 10M images
- for many recent datasets, training (but not running) a model is prohibitively costly.
- Solution : use off-the-shelf model, keep the representation extraction part, adapt the output layers for your own specific task $\Rightarrow$ finetuning
- One may want to finetune N models, then combine their decisions $\Rightarrow$ late fusion

Why do we need to do early fusion with deep networks?

Relying on multiple datasets

Dataset evolution in classification

- 1998 : MNIST, 10 classes and 60k+10k images
- 2004 : Caltech 101, 100 classes and 9k images
- 2009 : CIFAR 10, 10 classes and 60k images
- 2009 : ImageNet, 1000 classes and 10M images
- for many recent datasets, training (but not running) a model is prohibitively costly.
- Solution : use off-the-shelf model, keep the representation extraction part, adapt the output layers for your own specific task $\Rightarrow$ finetuning
- One may want to finetune N models, then combine their decisions $\Rightarrow$ late fusion

Why do we need to do early fusion with deep networks?

Relying on multiple datasets

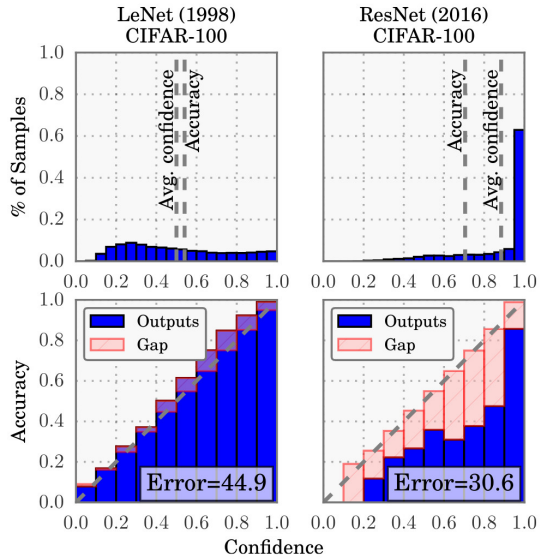
Dataset evolution in classification

- 1998 : MNIST, 10 classes and 60k+10k images
- 2004 : Caltech 101, 100 classes and 9k images
- 2009 : CIFAR 10, 10 classes and 60k images
- 2009 : ImageNet, 1000 classes and 10M images
- for many recent datasets, training (but not running) a model is prohibitively costly.
- Solution : use off-the-shelf model, keep the representation extraction part, adapt the output layers for your own specific task $\Rightarrow$ finetuning
- One may want to finetune N models, then combine their decisions $\Rightarrow$ late fusion

Late fusion

A decision/answer without the underlying uncertainty : very poor input for data fusion

Why is uncertainty so hard to obtain?



Score and confidence histograms for LeNet and ResNet-110 on CIFAR-100

Late fusion

A decision/answer without the underlying uncertainty : very poor input for data fusion

Why is uncertainty so hard to obtain?

- A central topic in the community
- Related to : confidence, robustness, reliability
- A major deterrent to deep learning adoption in critical applications
- Deep models get overconfident
- We need the posterior

Late fusion

A decision/answer without the underlying uncertainty : very poor input for data fusion

Why is uncertainty so hard to obtain?

- A central topic in the community
- Related to : confidence, robustness, reliability
- A major deterrent to deep learning adoption in critical applications
- Deep models get overconfident
- We need the posterior

Late fusion

A decision/answer without the underlying uncertainty : very poor input for data fusion

Why is uncertainty so hard to obtain?

- A central topic in the community
- Related to : confidence, robustness, reliability
- A major deterrent to deep learning adoption in critical applications
- Deep models get overconfident
- We need the posterior

Late fusion

A decision/answer without the underlying uncertainty : very poor input for data fusion

Why is uncertainty so hard to obtain?

- A central topic in the community
- Related to : confidence, robustness, reliability
- A major deterrent to deep learning adoption in critical applications
- Deep models get overconfident
- We need the posterior

Late fusion

A decision/answer without the underlying uncertainty : very poor input for data fusion

Why is uncertainty so hard to obtain?

- A central topic in the community
- Related to : confidence, robustness, reliability
- A major deterrent to deep learning adoption in critical applications
- Deep models get overconfident
- We need the posterior

Late fusion

Some possible approaches

- Bayes by backprop [Blundell et al., 2015] : costly
- MC Dropout [Gal and Ghahramani, 2016] : easy to deploy, costly for inference, patchy and sensitive
- Deep Ensembles [Lakshminarayanan et al., 2017] : easy to deploy, costly for training and inference, good performance
- TRADI [Franchi et al., 2020b] : tracking weight distributions using KF
- SLURP [Yu et al., 2021] : building auxiliary uncertainty estimators
- Uncertainty estimators based on some sort of clustering in the deep latent space
- One has to take into account that aleatoric and epistemic uncertainty are not equivalent!

Late fusion

Some possible approaches

- Bayes by backprop [Blundell et al., 2015] : costly
- MC Dropout [Gal and Ghahramani, 2016] : easy to deploy, costly for inference, patchy and sensitive
- Deep Ensembles [Lakshminarayanan et al., 2017] : easy to deploy, costly for training and inference, good performance
- TRADI [Franchi et al., 2020b] : tracking weight distributions using KF
- SLURP [Yu et al., 2021] : building auxiliary uncertainty estimators
- Uncertainty estimators based on some sort of clustering in the deep latent space
- One has to take into account that aleatoric and epistemic uncertainty are not equivalent!

Late fusion

Some possible approaches

- Bayes by backprop [Blundell et al., 2015] : costly
- MC Dropout [Gal and Ghahramani, 2016] : easy to deploy, costly for inference, patchy and sensitive
- Deep Ensembles [Lakshminarayanan et al., 2017] : easy to deploy, costly for training and inference, good performance
- TRADI [Franchi et al., 2020b] : tracking weight distributions using KF
- SLURP [Yu et al., 2021] : building auxiliary uncertainty estimators
- Uncertainty estimators based on some sort of clustering in the deep latent space
- One has to take into account that aleatoric and epistemic uncertainty are not equivalent!

Late fusion

Some possible approaches

- Bayes by backprop [Blundell et al., 2015] : costly
- MC Dropout [Gal and Ghahramani, 2016] : easy to deploy, costly for inference, patchy and sensitive
- Deep Ensembles [Lakshminarayanan et al., 2017] : easy to deploy, costly for training and inference, good performance
- TRADI [Franchi et al., 2020b] : tracking weight distributions using KF
- SLURP [Yu et al., 2021] : building auxiliary uncertainty estimators
- Uncertainty estimators based on some sort of clustering in the deep latent space
- One has to take into account that aleatoric and epistemic uncertainty are not equivalent!

Late fusion

Some possible approaches

- Bayes by backprop [Blundell et al., 2015] : costly
- MC Dropout [Gal and Ghahramani, 2016] : easy to deploy, costly for inference, patchy and sensitive
- Deep Ensembles [Lakshminarayanan et al., 2017] : easy to deploy, costly for training and inference, good performance
- TRADI [Franchi et al., 2020b] : tracking weight distributions using KF
- SLURP [Yu et al., 2021] : building auxiliary uncertainty estimators
- Uncertainty estimators based on some sort of clustering in the deep latent space
- One has to take into account that aleatoric and epistemic uncertainty are not equivalent!

Late fusion

Some possible approaches

- Bayes by backprop [Blundell et al., 2015] : costly
- MC Dropout [Gal and Ghahramani, 2016] : easy to deploy, costly for inference, patchy and sensitive
- Deep Ensembles [Lakshminarayanan et al., 2017] : easy to deploy, costly for training and inference, good performance
- TRADI [Franchi et al., 2020b] : tracking weight distributions using KF
- SLURP [Yu et al., 2021] : building auxiliary uncertainty estimators
- Uncertainty estimators based on some sort of clustering in the deep latent space
- One has to take into account that aleatoric and epistemic uncertainty are not equivalent!

Late fusion

Some possible approaches

- Bayes by backprop [Blundell et al., 2015] : costly
- MC Dropout [Gal and Ghahramani, 2016] : easy to deploy, costly for inference, patchy and sensitive
- Deep Ensembles [Lakshminarayanan et al., 2017] : easy to deploy, costly for training and inference, good performance
- TRADI [Franchi et al., 2020b] : tracking weight distributions using KF
- SLURP [Yu et al., 2021] : building auxiliary uncertainty estimators
- Uncertainty estimators based on some sort of clustering in the deep latent space
- One has to take into account that aleatoric and epistemic uncertainty are not equivalent!

References I

Bera, A., Galoppo, N., Sharlet, D., Lake, A. T., and Manocha, D. (2014).
Adapt: Real-time adaptive pedestrian tracking for crowded scenes.
In *2014 IEEE International Conference on Robotics and Automation, ICRA 2014, Hong Kong, China, May 31 - June 7, 2014*, pages 1801–1808. IEEE.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. (2015).
Weight uncertainty in neural network.
In *International conference on machine learning*, pages 1613–1622. PMLR.

Chari, V., Lacoste-Julien, S., Laptev, I., and Sivic, J. (2015).
On pairwise costs for network flow multi-object tracking.
In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5537–5545.

Dehghan, A., Modiri Assari, S., and Shah, M. (2015a).
Gmmcp tracker: Globally optimal generalized maximum multi clique problem for multiple object tracking.
In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4091–4099.

Dehghan, A., Tian, Y., Torr, P. H., and Shah, M. (2015b).
Target identity-aware network flow for online multiple target tracking.
In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1146–1154.

Franchi, G., Aldea, E., Dubuisson, S., and Bloch, I. (2020a).
Tracking hundreds of people in densely crowded scenes with particle filtering supervising deep convolutional neural networks.
In *Proceedings of the 27th International Conference on Image Processing (ICIP)*.

References II

Franchi, G., Bursuc, A., Aldea, E., Dubuisson, S., and Bloch, I. (2020b).
TRADI: tracking deep neural network weight distributions.
In Vedaldi, A., Bischof, H., Brox, T., and Frahm, J., editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XVII*, volume 12362 of *Lecture Notes in Computer Science*, pages 105–121. Springer.

Gal, Y. and Ghahramani, Z. (2016).
Dropout as a bayesian approximation: Representing model uncertainty in deep learning.
In *international conference on machine learning*, pages 1050–1059. PMLR.

Gordon, N. J., Salmond, D. J., and Smith, A. F. (1993).
Novel approach to nonlinear/non-gaussian bayesian state estimation.
In *IEE proceedings F (radar and signal processing)*, volume 140, pages 107–113. IET.

Isard, M. and Blake, A. (1998).
Condensation - conditional density propagation for visual tracking.
International journal of computer vision, 29(1):5–28.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. (2017).
Simple and scalable predictive uncertainty estimation using deep ensembles.
Advances in neural information processing systems, 30.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. (2017).
Focal loss for dense object detection.
In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

Neverova, N., Wolf, C., Taylor, G. W., and Nebout, F. (2016).
Moddrop: Adaptive multi-modal gesture recognition.
IEEE Trans. Pattern Anal. Mach. Intell., 38(8):1692–1706.

References III

Weinzaepfel, P., Revaud, J., Harchaoui, Z., and Schmid, C. (2013).
Deepflow: Large displacement optical flow with deep matching.
In *Proceedings of the IEEE international conference on computer vision*, pages 1385–1392.

Yu, X., Franchi, G., and Aldea, E. (2021).
Slurp: Side learning uncertainty for regression problems.
In *32nd British Machine Vision Conference, BMVC 2021, Virtual Event / November 22-25, 2021*.