Generative Models of Individuality for Fingerprints

Training data set

Minutiae $x$

Generative Model $p(x)$

Training

Sampling

EM Algorithm

Probability of Random Correspondence (PRC)
Sequential EM

Input: N training samples
Output: parameters of each of the K components

1. Initialize the distribution parameters
2. Repeat until convergence:
   1. E-Step:
      Estimate the expected value of the unknown variables, given the current parameter estimate.
   2. M-Step:
      re-estimate the parameters to maximize the likelihood of the data, given the estimates of the expectations of the unknown variables.
Basic Ideal of Parallel EM

- E step dominates the execution time on each iteration,
- E step is inherently data parallel if the parameters are available for each processor.
- The loop is parallelized by evenly distributing the data across the processors.
- Single Program Multiple Data (SPMD)
- Processor $P_0$ distributes the data by using MPI_Scatter.
- Call MPI_Allreduce to sum up the local parameters to obtain the new global parameters.
- Only one collective communication is needed in each iteration.
Parameter Estimation using EM

- **Minutiae distribution (Mixture of Gaussian and Von-mises)**

\[ p(x|\Theta) = \sum_{k=1}^{K} \pi_k \cdot \mathcal{N}(x|\mu_k, \Sigma_k) \cdot \mathcal{V}(\theta|\nu_k, \kappa_k) \]

**EM for Mixture of Gaussian and Von-mises**

Repeat until convergence:

**E step:**
\[ \gamma_{dk}^{(n+1)} = \frac{\pi_k \mathcal{N}(x_d | \mu_k, \sigma_k)}{\sum_{k=1}^{K} \pi_k \mathcal{N}(x_d | \mu_k, \sigma_k)} \]

**M step:**
\[ \pi_k^{(n+1)} = \frac{1}{D} \sum_{d=1}^{D} \gamma_{dk}^{(n)} \]
\[ \mu_{mk}^{(n+1)} = \frac{\sum_{d=1}^{D} \gamma_{dk}^{(n)} s_m}{\sum_{d=1}^{D} \gamma_{dk}^{(n)}} \]
\[ \nu_{mk}^{(n+1)} = \frac{1}{2} \tan^{-1} \left( \frac{\sum_{d=1}^{D} \gamma_{dk}^{(n)} \sin 2\psi_d}{\sum_{d=1}^{D} \gamma_{dk}^{(n)} \cos 2\psi_d} \right) \]
\[ I_0^{(n+1)}(\kappa_{mk}) = \frac{\sum_{d=1}^{D} r_{dk}^{(n)} \cos 2(\psi_d - \nu_{mk}^{(n+1)})}{\sum_{d=1}^{D} r_{dk}^{(n)}} \]
\[ \sum_{mk}^{(n+1)} = \frac{\sum_{d=1}^{D} \gamma_{dk}^{(n)} (s_m - \mu_{mk}^{(n+1)}) (s_m - \mu_{mk}^{(n+1)})^T}{\sum_{d=1}^{D} \gamma_{dk}^{(n)}} \]
Parallel EM

N input data (multi-dimensional)

Distribution by data
(MPI_Scatter operation)

Pre-compute
Intermediate parameters

Local parameters
computation

Global parameters
computation
(MPI_Allreduce operation)

E-Step

M-Step

Next iteration
#include "mpi.h"
……
int main(int argc, char *argv[]){
……
double gparam[K] = {...};
double lparam[K] = {...};
}
}
Initialize the global and local parameters

Initialize MPI
……
if (rank == MASTER) {
    fptr=fopen("minutiae.txt","r");
    fscanf(fptr,"%d %d %d", &x, &y, &t);
    MPI_Scatter(data, numpp, MPI_INT, localData, numpp, MPI_INT, 0, MPI_COMM_WORLD);
}
……
Read training data from disk

Iterate until convergence{
……
    for (idx=0; idx<nump; idx++)
        r[idx] = ...
………………………………………………………………………………………………………..
    for (idx=0; idx<numpp; idx++)
        localParam = ...
………………………………………………………………………………………………………..
    MPI_Allreduce(&localParam, &globalParam, K, MPI_DOUBLE, MPI_SUM, MPI_COMM_WORLD);
    globalParam = …
……
Parallel codes
E-Step
Sequential codes
M-Step
}
……
ierr = MPI_Finalize();
return 0;
}
Running time with varying numbers of processors

Input data set: 180,000 minutiae from NIST 4

Sweet spot:
T(40) = 263 s

Bottom point:
T(260) = 89 s
Input data set: 180,000 minutiae from NIST 4
Running Time Analysis

- **T(s):** running time of sequential codes
- **T(c_i):** parallelization costs with i processors (initialization, communication, etc.)
- **T(p_i):** running time of parallelized codes with i processors

Running time with n processors: \( T(n) = T(s) + T(c_n) + T(p_n) \)

- \( \Delta c_i = T(c_i) - T(c_{i-1}) \)
- \( \Delta p_i = T(p_{i-1}) - T(p_i) \)

Incremental running time:
\[
T(i) - T(i-1) = (T(s) + T(c_i) + T(p_i)) - (T(s) + T(c_{i-1}) + T(p_{i-1})) \\
= \Delta c_i - \Delta p_i
\]

If \( \Delta c_i < \Delta p_i \), running time decreases
If \( \Delta c_i > \Delta p_i \), running time increases

Sequential algorithm (one processor):
parallel part: 9556 s  >>  sequential part: 6 s

*Why the best running time is \( T(260) = 89 \) ?*
$\Delta p_i$ and $\Delta c_i$

Bottom point:
$T(260) = 89\, \text{s}$
Running time with varying amount of data

Experiment Setup: 20 processors
Results

Input data set: 180,000 minutiae from NIST 4