

MLBP: MAS for Large-Scale Biometric Pattern Recognition

Ram Meshulam Shulamit Reches Aner Yarden Sarit Kraus
{meshulr1,reches,yardena,sarit}@cs.biu.ac.il
Department of Computer Science
Bar-Ilan University, 52900
Ramat-Gan, Israel

ABSTRACT

Security systems can observe and hear almost anyone everywhere. However, it is impossible to employ an adequate number of human experts to analyze the information explosion. In this paper we present an autonomous multi-agent framework which, as an input, obtains biometric information acquired at a set of locations. The framework aims in real-time to point out individuals who act according to a suspicious pattern across these locations. The system works in large-scale scenarios. We present a scenario to demonstrate the usefulness of the framework. The goal is to point out individuals who visited a sequence of airports using face recognition algorithms. Simulation results show a high overall accuracy of our system in real-time.

Categories and Subject Descriptors

I.2.11 [Artificial Intelligence]: Distributed Artificial Intelligence

General Terms

Algorithms, Theory

Keywords

privacy, safety and security in agent systems, cooperative distributed problem solving in agent systems

1. INTRODUCTION

In this paper we address the general domain of Large-scale Biometric Pattern recognition (LBP). LBP problems include situations where there is a stream of biometric information acquired at a set of locations e.g., a set of cameras positioned in several airports. The goal of LBP problems is to point out individuals who act according to a predefined suspicious pattern in real-time. The combination of multiple locations and streams of input, inaccuracy of biometric tests [1] and real-time constraints make the LBP problems

hard to solve, and ideal candidate for MAS implementation [2]. A representative scenario is presented below.

1.1 Problem Description

In this paper we present a framework of a multi-agent solution for the LBP problems. We then present an implementation of this framework for a specific problem we call **The airport problem**. Assume a set of L airports denoted $ap_1, ap_2, \dots, ap_L$. Our goal is to identify and detain at airport L every passenger who flew from airport 1 to airport 2.... to airport L . At each airport there are cameras which take one picture of each passenger.

There are two boolean comparison algorithms, namely ca_1 and ca_2 . As an input each comparison algorithm receives two pictures. The comparison algorithm returns true if the pictures match, i.e. both pictures belong to the same person. Otherwise the algorithm returns false. We assume the algorithms are independent. Each comparison algorithm has a known *false acceptance rate* (FAR) which is the probability that the algorithm will receive two pictures which do not belong to the same person and it will return true. Each comparison algorithm has a known *false reject rate* (FRR) which is the probability that the algorithm will receive two pictures which belong to the same person and it will return false. We also know that the first algorithm has a lower FAR and the second algorithm has a lower FRR: $FAR_{ca_1} < FAR_{ca_2}$ and $FRR_{ca_2} < FRR_{ca_1}$. We define a *picture-set* of size k as a set of pictures taken from k airports. A key constraint of LBP is that the suspects do not need to cooperate with the system.

1.2 Difficulties and challenges

We would like to point out some of the attributes make LBP problems difficult. The examples refer to the airport problem, although all attributes exist in the LBP domain in general. Firstly, this is a large scale problem. Assume we would like to capture every person who visited two different airports ap_1 and ap_2 . Also assume there are 1000 people in each airport. The agent must create $1,000 \times 1,000 = 1,000,000$ picture-sets of size two and use a comparison algorithm to compare each pair of pictures. Note that the cameras' location can ensure that only pictures of passengers who came on a certain flight are taken, so 1000 images per airport is not an unrealistic scenario.

Moreover, biometric comparison algorithms are imperfect. For example, current state-of-the-art face recognition algorithms have an error margin that is unacceptable in large-scale situations [1]. Suppose all the pictures in the previous example are of different people. Even an algorithm with an

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

AAMAS'06 May 8–12 2006, Hakodate, Hokkaido, Japan.
Copyright 2006 ACM 1-59593-303-4/06/0005 ...\$5.00.

Algorithm 1 MLBP framework: local module

Local-module(Path p , item i , path q)

```
1 Let  $r = \mathbf{Phase1}(p, i, q)$ 
2 If  $r = \text{'borderline'}$  Then return  $\mathbf{Phase2}(q, i, q)$ 
3 return  $r$ 
```

Phase1(Path p , item i , path q)

```
1 Let  $q = \mathbf{AddItemToPath}(p, i, ca_1)$ 
2 Let  $m = \binom{L}{2} - LT$ 
3 If  $q^- > m$  Then return 'prune'
4 If  $((|q| < L) \text{ AND } (q^- \leq m))$  Then
4.1 Return 'continue'
5 If  $(q^- < m)$  Then return 'trigger alarm'
6 return 'borderline'
```

Phase2(Path p)

```
1 Let  $q$  be an empty path
2 Let  $m = \binom{L}{2} - HT$ 
3 For each item  $i$  in  $p.I$ 
3.1  $q = \mathbf{AddItemToPath}(q, i, ca_2)$ 
3.2 If  $q^- > m$  Then return 'prune'
4 return 'trigger alarm'
```

FAR of 1% will, on average, mistakenly positively identify 10,000 out of the 1,000,000 picture-sets. Suppose there is a person whose picture was taken at ap_2 . This picture is matched 1,000 times, each time against different pictures acquired at ap_1 . The probability that this picture will appear in a set that is falsely accepted is very high. Using this algorithm will make all passengers of the second airport be (probably falsely) detained.

2. MAS SOLUTION FRAMEWORK

We present a solution to the LBP problem assuming we have two biometric comparison-algorithms. The solution uses multiple agents to incrementally build candidate suspects. At each step, the algorithm tries to prune some of the candidates. Each agent consists of a local-module and an interaction-module.

The main information structure which is transferred between the agents is *path* defined as $p = \{I, R\}$. $I = \{i_1, i_2, \dots, i_k\}$ is a set of k items (e.g., face images of passengers). R is a set of the comparison results between any two items in I . Given a path p , we denote p^- as the number of negative comparison results in $p.R$. Similarly, we define p^+ as the number of positive comparison results in $p.R$. We also denote the number of items in $p.I$ as $|p|$. There are $p^+ + p^- = \binom{k}{2}$ comparison results in a path of length k . A path p and an item i can be joined together to create a new path of size $|p| + 1$. The method *AddItemToPath* gets a path p and an item i and returns a new path q . $q.I = \{p.I \cup i\}$ and $q.R$ holds $p.R$ along with the comparison results of items in $p.I$ and i .

Suppose we have a perfect comparison algorithm, with no FAR and FRR. The system should trigger an alarm only when it detects a path of size L which has the maximum number of positive comparison results, i.e. $p^+ = \binom{L}{2}$ and $p^- = 0$. For example, If $L = 3$, the system should trigger an alarm only if $ca(1, 2) = ca(1, 3) = ca(2, 3) = \text{true}$. In

actuality, however, comparison algorithms have FRR and we must consider cases where all items match, but some of the comparisons are falsely rejected. For the first comparison algorithm ca_1 we define a threshold LT where $0 \leq LT \leq \binom{L}{2}$. The algorithm triggers an alarm when a path of size L is created and $p^+ > LT$. In other words, the algorithm allows a path to have $p^- < \binom{L}{2} - LT$ false comparisons and still treats it as a path with matching items which might lead to a target person. Similarly we define a threshold HT for ca_2 .

Each agent is responsible for one biometric information source (e.g., a camera). An agent receives new information from two sources. The first source is from his biometric information unit (e.g., a passenger picture). The agent processes the acquired item and creates a path of size one. The second source of information is forwarded paths from other agents. Each agent holds a database of paths PDB in which it stores every new path it receives.

Algorithm 1 describes the local module of the MLBP framework. The local-module is embedded in each agent. As an input the module receives a path p and a biometric item i . It returns a string that classifies the joint path q which contains all items in p and i . The local module has three possible return values:

- 'prune' - There is no point in keeping the joint path.
- 'continue' - The joint path has a potential to become a target path.
- 'trigger alarm' The joint path is a target path and the system must notify about it.

The module may return 'continue' only if the size of the joint path is smaller than L . It may return 'trigger alert' only for full paths - paths of size L . Comparing a pair of pictures is time consuming. Thus, the algorithm uses ca_1 alone to classify the new path in most cases (lines 3-5 in function **Phase1**). The only scenario in which ca_2 is used is a borderline scenario. A borderline scenario occurs when a full path ($|q| = L$), has exactly LT positive results. The algorithm then uses ca_2 (function **Phase2**). First the algorithm creates a new path which contains all items in q , but this time it uses ca_2 to compare all pairs of items. The same classification rules are applied for q' , but this time the threshold is HT rather than LT .

Though ca_2 (which is used in phase 2 only) has a higher false alarm ratio, it is of less concern since at this point the number of paths has already been significantly reduced. Conversely, the low FRR of this algorithm makes it very effective in verifying the identity of the true suspect. Phase 2 is significantly more time consuming than phase 1. In phase 1 $p.R$ already contains comparison results of all items in p (using ca_1). The new item i is then compared with items in p , which totals $L - 1$ comparisons (worst case). In phase 2, the algorithm uses $\binom{L}{2}$ comparisons (worst case) to build a path from scratch using ca_2 .

Also note that the pruning made by the local-module is lossless. It is lossless in the sense that a system which uses the module will have the same accuracy with and without the pruning mechanism. This fact allowed us to ignore the pruning mechanism when we analyzed the algorithm performance in terms of accuracy. However, pruning is crucial in terms of run-time.

Algorithm 2 Airport Problem - Agent A_i

ProcessImage(Image im)1 For each Path p in PDB 1.1 Let q be empty path1.2 Let $r = \text{Local-module}(p, im, q)$ 1.3 If $r = \text{'trigger alarm'}$ Then trigger alarm1.4 If $r = \text{'continue'}$ Then forward q to A_{i+1}

L	N	LE	HE	$\overline{FRR}$	$\overline{FAR}$
4	500	1	2	0.0700	0.0011
4	500	2	0	0.0100	0.0682
5	500	2	3	0.0700	0.0022
6	500	2	3	0.1900	0.0000
4	1000	1	1	0.0967	0.0012
5	1000	2	3	0.0588	0.0040
6	1000	2	2	0.2225	0.0000
4	1500	1	0	0.1500	0.0001
4	1500	1	2	0.0700	0.0011

Table 1: Simulation results for selected scenarios

The local-module classifies a given path. The agents create paths, process them using the local-module and send them to the relevant agents. This flow of information is made according to the problem specifications. In the following subsection, we describe implementation of the interaction-module for the airport problem.

We have a camera installed in L airports: $ap_1, ap_2, \dots, ap_L$. At airport L we would like to detain every person who visited all airports *in a specific order*: $ap_1 \rightarrow ap_2 \rightarrow \dots, ap_L$. This allows us to use a simple sequential flow of paths as follows.

Agent A_1 (located at ap_1) acquires images and sends them to A_2 . The interior agent, A_k , $1 < k \leq L$ maintains a database of paths sent by A_{k-1} denoted PDB . Each path in PDB is of length $k-1$. When a passenger's image im is acquired at airport k , each path in PDB is sent to the local-module along with im (See algorithm 2). Paths that have enough positive comparison results are sent to A_{k+1} or trigger an alarm (if the path was created by the last agent).

We are interested in individuals who traveled through the airports in a specific order. This allows the agent to restrict the paths' flow from agent A_i to agent A_{i+1} . Moreover, the costly procedure **Phase2** of the local-module is activated only by the last agent, which is the only one to process paths of size L .

3. EXPERIMENTAL RESULTS

We would like to know how the problem size (number of people N and number of information sources L) affects the accuracy of the algorithm. We also want to know how these parameters affect the running time of the algorithm. We would like to verify that the algorithm runs in real-time.

In order to answer these questions we implemented a simulation of the airport problem. For statistical significance we averaged results from 100 trials for each parameter studied. For each trial, we created a database which contains the flight plan for each passenger. We assumed a distribution function where most of the passengers only visit one or two airports out of the possible L airports. We modeled this behavior as an exponential decay function with the fewest people visiting all L airports.

We then ran experiments on various scenarios to test the overall behavior of the system. Table 3 shows various problem instances and the global FRR and FAR achieved by the MLBP algorithm. The first two columns describe the number of airports and the number of pictures taken at each airport. Columns three and four are the thresholds chosen for the simulation. The last two columns show the actual FRR and FAR. The results are very encouraging. For example, the results in line 5 shows that out of 1000 passengers, only one passenger will need special attention and the system will point out a target suspect with a reliability of 93%. We also tried to use ca_2 in the first phase and ca_1 in the second phase of the local-module (with $L = 1000$). The result was $\overline{FAR} = 1$! The high $\overline{FAR}$ of ca_2 caused each passenger in ap_L to have at least one path with enough false positive classifications to trigger an alarm.

We also were successful in significantly reducing the time complexity factor of this problem. We focused on the last agent since its time complexity is the most significant. The last agent is the only agent who uses the costly second pruning phase. This agent has the most time intensive decision to make, deciding in real time which passenger to single out. Other agents have the luxury of having more time before a final decision has to be rendered. Even in the worse case we studied, when $N = 1000$ and $L = 5$, the final agent was able to render a judgment under 12.5 seconds using a Pentium IV 3.0 Ghz computer with 1 GB of memory.

It is important to note that increasing the length of the path (L) does not necessarily increase the time complexity. The overall time needed by the last agent to make a decision is affected by two factors: 1) LT : the first threshold determines the number of paths reaching the last agent. A low threshold means a large number of paths that reach the last agent. 2) L : Borderline paths go through the second phase. The algorithm uses $\binom{L}{2}$ tests (in the worst case) on the second phase. Therefore, the longer the path the more time needed by the last agent for each path.

4. CONCLUSIONS

In this paper we addressed the domain of large-scale biometric pattern-recognition. The combination of multiple locations and streams of input, inaccuracy of biometric tests and real-time constraints make LBP problems hard to solve, and ideal candidates for MAS implementation.

We have presented the MLBP - a multi-agent solution to handle the LBP problem. The solution uses multiple agents to incrementally build solution candidates. Each agent consists of a local-module and an interaction-module. The local-module prunes candidates which do not reach a certain threshold. Borderline cases go through a secondary process of classification. *A web version which contains additional domains and a full theoretical analysis is available at www.cs.biu.ac.il/~meshulr1/MLBP.pdf*

Acknowledgements : This work was supported by MAFAT and ISF #8008. In memory of Matat Rosenfeld-Adler.

5. REFERENCES

- [1] A. K. Jain, A. Ross, and S. Prabhakar. Intro. to biometric recognition. *Transactions on Circuits and Systems for Video Technology*, 14(1), 2004.
- [2] P. Weinstein, H. V. Parunak, P. Chiusano, and S. Brueckner. Agents swarming in semantic spaces to corroborate hypotheses. In *Proc. of AAMAS04*, 2004.