MLBP: MAS for Large-Scale Biometric Pattern Recognition

Ram Meshulam, Shulamit Reches, Aner Yarden, and Sarit Kraus

Department of Computer Science, Bar-Ilan University, 52900, Ramat-Gan, Israel
{meshulr1,reches,yardena,sarit}@cs.biu.ac.il

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 two scenarios to demonstrate the usefulness of the framework. The goal in the first scenario is to point out individuals who visited a sequence of airports, using face recognition algorithms. The goal in the second scenario is to point out individuals who called a set of phones, using speaker recognition algorithms. Theoretical performance analysis and simulation results show a high overall accuracy of our system in real-time.

1 Introduction

In this paper we address the general domain of **L**arge-scale **B**iometric **P**attern recognition (LBP). The problems in the LBP domain 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 domains 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 [8] and real-time constraints make the LBP problems hard to solve, and they are ideal candidates for MAS implementation [15]. In the age of global terrorism, LBP has become even more important. Two representative scenarios which fit the LBP domain are presented below.

Given a stream of face images acquired at a set of airports, we would like our system to point out, in real-time, any person who visits a set of airports in a certain predefined order. The alarm must be triggered as soon as the suspect's image is acquired detected at the last airport(s). For example: we would like to detain anyone who flew from city A or B or C to D or E and then to F or G. In this example, a passenger who flies from A to D and continues to F (without any stop) should trigger an alarm as soon as she lands in F. Another sample scenario is a high security facility heavily equipped with cameras. We define a prohibited sequence of movements between locations in the facilities. For example, from the safe-room to the locker-room and to the parking lot (although going to the cafeteria and then to the locker room and parking lot is allowed). We would like the system to point out in real time any individual who traveled the prohibited route.

1.1 Problems 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 two specific problems:
The airport problem - Assume a set of L airports denoted $ap_1, ap_2, ..., ap_L$. Our goal is to identify and detain at airport L every passenger who flew from airport 1 to airport 2.... to airport L. We denote $x \rightarrow y$ as flying from airport x to airport y. At each airport there are cameras which take one picture of each passenger.
The tapping problem - Assume an autonomous system which listens to L phones. The goal of the system is to identify in real time anyone who called all L phones within a specified period of time (e.g. a week), with no specific order. The system should trigger an alarm as soon as the last call ends. Note that the caller might call from several different phones.

For each biometric domain, we apply two different boolean comparison algorithms, namely ca_1 and ca_2. As input, each comparison algorithm receives two biometric items (a pair of pictures or a pair of call recordings). The comparison algorithm returns true if the items match, i.e. both items 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 receives two items which do not belong to the same person and it returns true. Each comparison algorithm has a known *false reject rate* (FRR) which is the probability that the algorithm receives two items which belong to the same person and it returns false. There is a trade-off between FAR and FRR. Without loss of generality, assume that the first comparison-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 an *item-set* of size k as a set of items taken from k airports or phones. A key constraint of LBP is that the suspects do not actively cooperate with the system (as opposed to fingerprint or iris-scanning tests which require the 'suspect' to cooperate).

1.2 Difficulties and challenges

We would like to point out some of the attributes that make LBP problems difficult. The examples refer to the airport problem, although all attributes exist in the tapping problem as well and in the LBP domain in general. First, 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' locations ensure that only pictures of passengers who arrived on a certain flight are taken, so 1000 images per airport is not an unrealistic scenario.

Moreover, biometric comparison algorithms are not perfect. For example, current state-of-the-art face recognition algorithms have an error margin that is unacceptable in large-scale situations [8]. Suppose all the pictures in the previous example are of different people. Even an algorithm with a 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 appears in a set that is falsely accepted is very high. Using this algorithm will cause all the passengers of the second airport to be (probably falsely) detained.

Another problematic attribute is the influence of a target person (a person who visited the desired airports) on the FAR of innocent people. Suppose we are interested in a pattern of three airports. Assume that an individual c has actually visited airports $1 \rightarrow 2 \rightarrow 3$. Also assume there were 1000 visitors at each airport. In this case there are $1,000$ picture-sets at the third airport out of 1000 in which two pictures of the first two airports belong to c and the third picture is a picture acquired at airport 3. There are 3 possible picture comparisons in each picture-set $\langle 1,2 \rangle, \langle 1,3 \rangle$ and $\langle 2,3 \rangle$ (we denote a pair of one image acquired at airport x and one image acquired at airport y as $\langle x,y \rangle$). For each person visiting airport 3 there is one picture-set which contains her picture (picture 3) along with two matching pictures (pictures 1 and 2). The chance that such a picture-set will be falsely classified as positive is larger than a picture set where all the pictures belong to different individuals.

To overcome the above challenges and provide real-time solutions, we present the **M**ulti-agent system for the **LBP** (MLBP). As the name suggests, this is a framework of a multi-agent system, aimed to solve LBP problems. Two main reasons have encouraged us to choose a multi-agent approach. First, the real-time constraint combined with the large-scale factor create a need to distribute the problem and parallelize it among multiple computational units. Second, the LBP involves separate 'stations' or sources, of biometric data (e.g. phones or cameras). It is only natural to put an agent in charge of each source, handling initial acquisition, preprocessing etc. and communicating with other agents to produce the final output. Each agent holds a partial solution database and creates new candidates based on new information. The agent locally prunes candidates out of the system as much as possible without losing important knowledge.

The paper is structured as follows. Section 2 contains the solution framework divided into two. The first part, called the local-module is responsible of classifying and pruning partial candidates. The local-module code is the same for all LBP problems. We describe the module and provide a theoretical analysis. The second part of the MLBP framework is the interaction-module which is a code that creates new solution candidates and uses the local-module to classify them. The interaction-module code is problem-specific. We present an implementation of the interaction-module for the airport problem and the tapping problem. Sec-

Algorithm 1 MLBP framework: local module

```
Local-module(Path $p$, item $i$)
1 Let $r$,$q$=Phase1($p$, $i$)
2 If $r$=‘borderline’ Then return Phase2($q$, $i$)
3 return $r$,$q$

Phase1(Path $p$, item $i$)
1 Let $q$ =AddItemToPath($p$, $i$, $ca_1$)
2 If $q^- > LE$ Then return ‘prune’,$null$
3 If ($|q| < L$) Then
3.1 return ‘continue’,$q'$
4 If ($q^- < LE$) Then return ‘trigger alarm’,$q'$
5 return ‘borderline’,$q'$

Phase2(Path $p$)
1 Let $q$ be an empty path
2 For each item $i$ in $p.I$
2.1 $q$=AddItemToPath($q$, $i$, $ca_2$)
3.2 If $q^- > HE$ Then return ‘prune’
4 return ‘trigger alarm’,$q$


AddItemToPath(Path $p$, item $i$, comparisonAlg $CA$)
1 Let $p' = p$
2 For each item $i'$ in $p'$
2.1 add $i$,$i'$,CA($i$, $i'$) to $p'.R$
3 Add $i$ to $p.I$
4 return $p'$
```

tion 3 describes simulation results which are matched against the theoretical analysis. In section 4 we present an overview of related problems and domains.

The LBP problem lies at the junction of well known domains [12, 8, 3, 5]. We mention each domain and point out the differences between it and the LBP domain. A discussion and summary are presented at the end of the paper.

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.

2.1 Solution framework

The main information structure which is transferred between the agents is *path* defined as $p = \{I, R\}$.

$I = \{i_1, i_2, ...i_k\}$ is a set of k items (e.g. face images or call recordings). 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 $p.R$. We also denote the number of items in $p.I$ as $|p|$. There are $\binom{k}{2}$ 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* (Algorithm 1) 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 between 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 p of size L which has the maximum number of positive comparison results, i.e. $p^- = 0$. For example, If $L = 3$, the system should trigger an alarm only if $ca(1, 2) = ca(1, 3) = ca(2, 3) = true$. In actuality, however, comparison algorithms have FRR and we must consider cases where all items are of the same person, but some of the comparisons are falsely rejected. For the first comparison algorithm ca_1 we define a threshold LE where $0 \leq LE \leq \binom{L}{2}$. The algorithm triggers an alarm when a path of size L is created and $p^- < LE$. In other words, the algorithm allows a path to have $p^- < LE$ false comparisons and still treats it as a path with matching items which might lead to a target person. Similarly we define a threshold HE for ca_2.

Each agent is responsible for one biometric information source (e.g. a phone). An agent receives new information from two sources. The first source is from his biometric information unit (e.g. a phone call recording). 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 the joint path q which contains all items in p and i and a string which classifies q. The local module has three possible return values:

– 'prune' - There is no point in keeping the joint path (and the returned path is *null*).
– '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 alarm' 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 LE negative results. The algorithm then uses ca_2 (function **Phase2**) as follows. First the algorithm creates a new path q' which contains all items of q. Then it uses ca_2 to compare all pairs of items. The same classification rules are applied for q', but this time the threshold is HE rather than LE.

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 one item is added. The new item i is compared with items in p, which totals $L-1$ comparisons (worst case). In phase 2, the algorithm matches all pairs of pictures using ca_2, which totals to $\binom{L}{2}$ comparisons.

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.

Note that the same picture might appear in many paths. For example, Agent 3 gets paths of size 2 which differ only by picture 2. Thus, the algorithm compares pairs of pictures over and over again. For example, consider two paths $\langle im_1, im_2\rangle$ and $\langle im_1, im_3\rangle$ forwarded to airport 3. Suppose a picture im_4 was acquired at airport 3. The agent will be required to match the following pairs of pictures:$\langle im_1, im_4\rangle$, $\langle im_2, im_4\rangle$ for the first path and $\langle im_1, im_4\rangle$, $\langle im_3, im_4\rangle$ for the second path. Instead of activating the comparison algorithm again, the results can be cached. The outcome of the above is that the main time-consuming action of the MLBP algorithm is table *lookup* rather than actual *item-comparison.* This notion is important for the time performance analysis discussed in section 3.

2.2 Performance Analysis

The FAR and FRR of the comparison algorithms are known in advance. However, a user of such a system will be interested in the *global* FRR and FAR, denoted $\overline{FRR}$ and $\overline{FAR}$, respectively. Referring to the airport problem, for example, one would like to know 1) What are the chances that a suspect who visited all airports will not be detained at the last airport; 2) What are the chances that a person who did not visit all airports will be falsely detained.

The thresholds LE and HE affect the relationship between $\overline{FRR}$ and $\overline{FAR}$: choosing high thresholds will increase the number of innocent people detained, and decrease the number of target people who get away. Choosing low thresholds will have the opposite effects.

The following paragraphs describe a formula which expresses the relationship between $\overline{FRR}$, $\overline{FAR}$, LE and HE. The formula allows users of the MLBP system to determine $\overline{FRR}$ and $\overline{FAR}$ through LE and HE adjustments.

In order to estimate the value of $\overline{FRR}$, we find the lower bound of $1-\overline{FRR}$, i.e. we calculate the probability that target suspect x will be detained using only the first phase. In other words, what is the probability that at least one path containing a biometric item of x in its last stage will have exactly LE negative results or less?

To simplify our formula, we assume all individuals except x do not "appear" in more than one location (e.g. in the tapping problem, all callers beside x made only one phone call each). This assumption gives us a lower bound for $1-\overline{FRR}$ and therefore an upper bound for $\overline{FRR}$. This is based on the fact that $1-FRR > FRR$, thus the probability of having a path with at most LE negative results is larger when several individuals appear in more than one location.

Step I We will calculate the probability of a specific path to have exactly LE negative results.

The number of tests in a full path of length L is $F(L)=\frac{L(L-1)}{2}$. The probability for exactly K positive results in a path which contains only the items of x is :

$$A_0^K = \binom{F(L)}{K}(1-FRR)^K FRR^{F(L)-K} \ .$$

The probability that a path with $L-j$ items of x and j items of different objects will contain exactly K positive results is:

$$A_j^K = \sum_{i=0}^{F(L)-K} \binom{S_j}{i} FAR^{S_j-i}(1-FAR)^i \cdot \\ \binom{F(L)-S_j}{K-(S_j-i)}(1-FRR)^{K-(S_j-i)} FRR^{F(L)-K-i} \tag{1}$$

where $S_j = \{ \begin{matrix} 0 & j=0 \\ \sum_{i=1}^{j}(L-i) & j\neq 0 \end{matrix}$

Step II Let a_i^j be the probability for exactly j positive results in a path with $L-i$ items of x and i items of different objects in the second pass. We will calculate a_i^j exactly as we calculated A_i^j, except for the values of FRR and FAR which are different in the first and the second passes. Thus the probability that a path with j items different from x and $L-j$ items of x has exactly LE negative results and will also pass the second path is:

$$A_j^{F(L)-LE} \sum_{i=F(L)-HE}^{F(L)} a_j^i \ .$$

To sum up the probability for such a path to pass the two passes is

$$C_j = A_j^{F(L)-LE} \sum_{i=F(L)-HE}^{F(L)} a_j^i + \sum_{i=F(L)-LE+1}^{F(L)} A_j^i .$$

Step III Now we will calculate the probability that at least one path will pass the two passes. There is only one path in which all the items are only of x.

Denote

$$B_0 = C_0 \ , \ F_i = \binom{L-1}{i} .$$

There are $(N-1)^i F_i$ different paths with $L-i$ items of x and the other i items are different from each other. We then find that the probability that at least one path of this form will pass the two passes is:

$$B_i = 1 - (1 - C_i)^{(N-1)^i F_i} \ . \tag{2}$$

The final formula for the lower bound of $1 - \overline{FRR}$ is:

$$1 - \prod_{i=0}^{L-1} (1 - B_i) \tag{3}$$

So the upper bound of $\overline{FRR}$ is:

$$\prod_{i=0}^{L-1} (1 - B_i) \tag{4}$$

Now we are interested in calculating $\overline{FAR}$, i.e. the probability that an individual who is not a target will be detained because a path in which the last item was acquired falsely triggered the alarm. We consider a group of individuals which appears in exactly $m < L$ locations, and we want to calculate the probability that a specific item will falsely trigger the alarm. We then multiply this result by the a priori probability that an individual appears in m locations.

In order to find an upper bound for $\overline{FAR}$ we consider the path which gives us the largest probability of having maximum positive results. Let us refer to the case in which the items are distributed throughout all the information sources they match, i.e. that every item in the path is located in m information sources. The probability that such a path will contain exactly K positive results for $m > 1$ is:

$$D_m^K = \sum_{i=0}^{F(L)-K} \binom{f}{i} (1 - FRR)^{f-i} FRR^i \binom{F(L) - f}{K - (f - i)} FAR^{K-(f-i)} (1 - FAR)^{F(L)-(K+i)} \tag{5}$$

where $f = \binom{m}{2} \lfloor \frac{L}{m} \rfloor + \binom{L \mod m}{2}$

For $m = 1$, $D_1^K = \binom{F(L)}{K} FAR^K (1 - FAR)^{F(L)-K}$. The probability that such a *Path* has exactly LE negative results and also passes the second path is:

$$D_m^{F(L)-LE} \sum_{i=F(L)-HE}^{F(L)} d_m^i \ ,$$

where d_m^i is the probability for such a $Path$ to have exactly $F(L) - i$ negative results in the second pass, which means the same as D_m^i, except for the values of FRR and FAR which are different in the second pass.
In conclusion, the probability that (in a situation where every item exactly matches $m < L$ biometric information sources) a path will pass the two passes is:

$$G_m = D_m^{F(L)-LE} \sum_{i=F(L)-HE}^{F(L)} d_m^i + \sum_{i=F(L)-LE+1}^{F(L)} D_m^i .$$

Therefore the upper bound for the probability that at least one path will pass the two passes, when every item matches $m < L$ information sources is

$$E_m = 1 - (1 - G_m)^{N_m} \ . \tag{6}$$

where:

$$N_m = N^{L-m-1}(\binom{L-1}{m} y p_L + \sum_{i=0}^{L-m-1} \binom{L-i-1}{m} y p_{L-i-1} \\ + \sum_{i=0}^{L-m-1} \binom{L-i-1}{m} N^{1-i}) \tag{7}$$

$N_1 = N^{L-1} - \sum_{m=2}^{L-1} N_m$.
is an upper bound for the number of paths with m equal items, y is the total number of items and p_i is the a priory probability that an item appears in i information sources. Because of the linear dependency between the paths, this formula provides an upper bound. So the final formula for the upper bound of $\overline{FAR}$ is:

$$1 - \prod_{m=1}^{L-1} (1 - E_m) \ , \tag{8}$$

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 paragraphs, we describe implementations of the interaction-module for the airport problem and the tapping problem.

2.3 Interaction-module

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

Algorithm 2 Airport Problem - Agent A_i

```
$ProcessImage$(Image $im$)
1 For each Path $p$ in $PDB$
1.1 Let $r, q$=**Local-module**($p$, $im$)
1.2 If $r$=='trigger alarm' Then trigger alarm
1.3 If $r$=='continue' Then forward $q$ to $A_{i+1}$
```

Algorithm 3 Tapping problem - agent A_i

```
**IncomingCall**(Call $call$)
1 add $call$ to calls database $C$
2 For each path $p$ in $PDB$
2.1 Let $r$,$q$=**Local-module**($p$,$c$) 2.2 If $r$=='trigger alarm' Then trigger alarm
2.3 If $r$=='continue' Then forward $q$ to agent $A_{i+1}$

**IncomingPath**(Path $p$)
1 add $p$ to $PDB$
2 For each call $c$ in $C$
2.1 $r$,$q$=**Local-module**($p$,$c$)
2.2 If $r$='continue' Then
2.2.1 forward $q$ to agent $A_{i+1}$
```

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).

Tapping Problem: Here, the goal is to locate a caller who phoned all L phones tapped by our system in real-time. Compared to the airport problem, the tapping problem is harder to solve. Current state-of-the-art speaker recognition algorithms are far less accurate than face recognition algorithms [11]. Moreover, the tapping problem does not require a specific order of calls, and any order of calls must be considered (we enumerate the phones from 1 to L arbitrarily). Algorithm 2.3 describes the implementation of the MLBP interaction-module for the tapping problem embedded in agent i.

The interaction-module of the tapping problem is similar to the airport interaction-module with one main difference. A tapping agent which gets a path from a previous agent must match it with all calls acquired earlier (see function **IncomingPath** in algorithm 2.3). The agent must do this because in most cases, the order of incoming calls is different from the order of phones. For example, suppose a person called phones 2,3 and then 1. Agent 2 and 3 will do nothing until the matching call was acquired by agent 1. Agent 1 will forward the call to agent 2. Agent 2 must match the call with previous calls to discover if there is a matching call. The goal in the airport problem was to locate a sequential order of airport travel, thus matching incoming paths with previous items was unnecessary.

Given the same initial parameters, both airport and tapping algorithms will produce the same paths and return the same results since both algorithms forward a path to the next agent iff it has not exceeded the allowable number of false matches. Note that 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

In this section we wanted to answer the following questions: (1) how the problem size (number of people N and number of information sources L) affects the accuracy of the algorithm, (2) how these parameters affect the running time of the algorithm, (3) how close the theoretical analysis predictions are to simulation results.

In order to answer these questions we implemented simulations of the airport problem and the tapping problem. For statistical significance we averaged results from 100 trials of each parameter studied. For each trial of the airport problem, 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 made similar preparations for the tapping problem simulation.

3.1 The airport problem

We studied scenarios of 4,5 and 6 airports. In each airport we assumed 500, 1000 and 1500 pictures of passengers. For each scenario, we experimented with increasing values of LE and HE. We stopped when the resulting FAR was no longer reasonable. We used the performance rates of the state-of-the-art face recognition algorithms [10] as comparison algorithms ca_1 and ca_2. ca_1 has a 10%/1% FRR/FAR. ca_2 has a 4%/10% FRR/FAR.

Table 3.1 shows selected simulations and the theoretical bounds[1]. Each line represents an average of 100 simulations over one scenario using a certain set of parameters. The first and second columns show the number of airports and the number of pictures taken in each airport respectively. The third column shows the number of false matches we allow for a path to have and still consider it as a target path. There are two numbers for the first and second phase of the algorithm. Columns 4-7 shows the simulations results (columns 4-5) and

[1] We have decided to discuss our results using a table and avoid a graphical representations such as a ROC curve. There is no meaning to a continuous line representation when the number of possible results is small (it is the number of assignments to the LE and HE thresholds).

			Simulation		Analysis	
L	N	LE,HE	$\overline{FRR}$	$\overline{FAR}$	$\overline{FRR}$	$\overline{FAR}$
4	500	1,0	0.1500	0.0001	0.1900	0.0010
4	500	1,1	0.0900	0.0002	0.1200	0.0001
4	500	1,2	0.0700	0.0011	0.1050	0.0004
4	500	2,0	0.0100	0.0682	0.0230	0.0710
5	500	2,3	0.0700	0.0022	0.0690	0.0004
5	500	2,4	0.0650	0.0055	0.0580	0.0010
6	500	0,1	0.8400	0.0000	0.8180	0.0000
6	500	1,2	0.4700	0.0000	0.4570	0.0000
6	500	2,3	0.1900	0.0000	0.1840	0.0000
4	1000	1,1	0.0967	0.0012	0.1180	0.0010
4	1000	1,2	0.0900	0.0037	0.0894	0.0019
5	1000	2,3	0.0588	0.0040	0.0490	0.0003
5	1000	2,4	0.0575	0.0108	0.0230	0.0010
6	1000	2,1	0.2500	0.0000	0.2150	0.0000
6	1000	2,2	0.2225	0.0000	0.1890	0.0000
4	1500	1,0	0.1500	0.0001	0.1940	0.0035
4	1500	1,1	0.0900	0.0002	0.1200	0.0040
4	1500	1,2	0.0700	0.0011	0.0940	0.0088

Table 1. Airport problem - Various scenarios and results.

the theoretical predictions (columns 6-7) for each scenario. Performance is measured in terms of FRR and FAR. Though the table shows only a portion of the simulations, the following paragraphs refer to all data collected.

In most cases, the theoretical results have quite accurately predicted the results obtained in simulations. The differences between simulations results and analysis have not exceeded 6% for both $\overline{FAR}$ and $\overline{FRR}$. In many cases it was less than 1%. This implies that the user may use the theoretical analysis to choose a desired $\overline{FRR}$ and $\overline{FAR}$ by assigning the thresholds LE and HE without actually running any experiments. For example, given a set of 4 airports and 500 pictures taken at each airport, one may choose to assign $LE = 1$, $HE = 2$ (line 1 in table 3.1). The outcomes of such an assignment risk a 15% false rejection rate while avoiding the need to detain innocent passengers. Alternately, assigning $LE = 2$, $HE = 0$ (line 4) might be chosen to decrease the false rejection rate to 1% while increasing the false alarm rate to 6.8%. This means that about 34 passengers will need manual examination.

3.2 The tapping problem

We studied scenarios of 4 and 5 phone lines. We assumed 100 or 200 calls from each line which is a reasonable estimation for the number of calls made in several days. We used state-of-the-art relevant speaker recognition algorithms (two speaker conversations with limited data) as ca_1 and ca_2 [11]. ca_1 has a 20%/5% FRR/FAR. ca_2 has a 15%/10% FRR/FAR.

			Simulation		Analysis	
L	N	LE,HE	$\overline{FRR}$	$\overline{FAR}$	$\overline{FRR}$	$\overline{FAR}$
4	100	1,2	0.2900	0.0206	0.2510	0.0199
4	100	1,3	0.2400	0.0473	0.1110	0.0512
4	100	1,4	0.1900	0.1693	0.0680	0.1990
5	100	2,4	0.3800	0.0070	0.0800	0.0140
5	100	3,3	0.1300	0.1373	0.0001	0.1950
6	100	4,3	0.2100	0.0045	0.0186	0.0222
6	100	4,2	0.2100	0.0038	0.0380	0.0152
6	100	4,3	0.2100	0.0045	0.1860	0.0222
4	200	1,1	0.3400	0.1137	0.2750	0.1160
4	200	1,2	0.2800	0.1302	0.1600	0.1330
5	200	2,4	0.2600	0.0347	0.0210	0.0410
5	200	2,5	0.2400	0.0401	0.0130	0.0460
6	200	4,3	0.1400	0.0430	0.0060	0.0570
6	200	4,2	0.1500	0.0380	0.0077	0.0480

Table 2. Tapping problem problem - Various scenarios and results.

The number of phone calls of the tapping problem is less than the number of pictures taken in the airport problem. However, the comparison-algorithms are less accurate. Thus, reaching a reasonable global FRR and FAR is harder. Table 3.2 shows selected simulations and the theoretical bounds. Each line represents an average of 100 simulations over one scenario using a certain set of parameters. The table structure is similar to table 3.1.

Here too, the theoretical analysis has successful predictions of the $\overline{FAR}$. However, the FRR predictions of the theoretical analysis sometimes do not match the results obtained by simulations. The $\overline{FRR}$ measures the relative number of rejections out of a total number of target persons. For the tapping problem, we assumed that the number of people calling all phones is only one or two. This means that the number of possible values for each simulation is extremely low. For example, if the scenario includes one target person, the possible values of the catching rate are either 0% or 100% . The theoretical analysis which assumes a valid statistical domain fails to predict such a scenario. In spite of the high inaccuracy of voice-recognition algorithms, the algorithm produced reasonable rates in various scenarios. Note that a relatively high percentage of FAR is still acceptable since the tapping problem deals with a relatively small number of items (100 and 200 in our simulations).

We also tried to use ca_2 for the first phase and ca_1 for the second phase of the local-module (with $L = 1000$). The result was $\overline{FAR} = 1$! Assigning these values in the formula confirmed the results. The high FAR of ca_2 led to the result that each passenger in the last airport had at least one path which had enough false positive classifications to trigger an alarm.

We also measured the time performance of the MLBP algorithm. As mentioned in subsection 2.1, the comparison results are cached. Thus, we do not execute a comparison algorithm more than once on a pair of items. The time performance is dominated by the number of matches lookups. Using a comparison-algorithm with high FAR in phase I of the local-module will generate many paths. Obviously, a high number of items N also increases the number of paths. A high number of biometric sources L influences mainly the last agent. 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. The algorithm uses $\binom{L}{2}$ tests (in the worst case) in the second phase. Therefore the longer the path, the more time is needed by the last agent for each path.

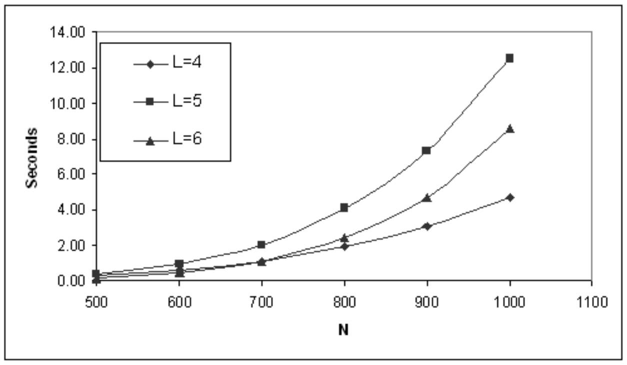

Fig. 1. Time of last agent per N,$L = 5$

All tapping simulations ended in a matter of seconds, so we focus our analysis on the airport problem. Figure 1 shows time performance of the last agent for 4,5 and 6 airports over increasing number of pictures taken at each airport. When $L = 4$, $LE = 2$, when $L = 5$, $LE = 2$ and when $L = 6$, $LE = 1$. Note that even 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.

Two interesting phenomena arise from figure 1: (1) five-airport simulation is slower than six-airport simulation. (2) Moreover, below 700 pictures four-airport simulation is slower than six-airport. Both phenomena can be explained considering the previous paragraph. As mentioned above, LE has a large influence on time performance. Only one error was allowed for the six-airport simulation while the five (and four) airport simulations had two errors allowed. Thus, more paths were generated and forwarded in the five-airport simulation. The reason the six-airport simulation became slower than four-airport simulation as the number of pictures increased is the increasing number of paths reaching phase II. Phase II, which is executed on borderline paths of size L, involves $L(L-1)/2$ picture matches which are 6 matches for four-airport simulation and 15 matches for six-airport simulation. As the N grows, so does the number of borderline cases, which slowed down the six-airport simulation.

4 Related Work

The LBP problem lies at a junction of well known domains: multi-modal biometric identification [12, 8], data mining [3, 5] and bioinformatics [2]. In the following paragraphs we will present a brief overview of these subjects and explain the LBP domain's uniqueness. We use the airport problem as a representative problem of the domain.

The solutions to all LBP problems mentioned above must involve a biometric system [8]. A biometric system is a pattern recognition system. It acquires biometric data from a target person. A feature set is then extracted from the acquired data. The feature set is compared to known feature sets residing in a database. One sub-domain of biometrics is biometric *identification* [14]. A biometric identification system tries to identify an individual by matching his/her biometric feature set to all feature sets in its database. Three main differences make it impossible to use popular identification biometric algorithms to solve LBP problems. First, there is no template database to which the acquired picture can be compared. The person who traveled through L airports in a sequence was not known to the system a priori. His recent actions made him a target. Second, there is not a single test which can classify a person as a target. Even if the biometric tests were perfect, only a series of tests with positive results makes a person a target. For example, only a sequence of $L - 1$ comparisons of L pictures taken at L airports will indicate that the same person has traveled through the desired airports. The last difference is the time issue. The results must be calculated in real-time, before the suspect leaves the last airport.

The local-module uses two phases of processing to classify paths. *Multi-modal* biometric systems use multiple biometric modalities [12, 8, 13]. Using more than one type of evidence or algorithm can improve robustness to noise, and increase security and accuracy. The tests are either merged together or used in an ordered way as a main and a secondary classifier. For example, in [13] iris and face biometric data are jointly used to achieve a higher accuracy of identity recognition. Hong and Jain developed multi-modal systems which combine face and fingerprint information [7]. Face recognition was first used to create the best n possible matches. They chose to use this method first due to its speed. Then, fingerprint information was used to identify the person from the n possible matches. To solve the LBP problem we use two comparison algorithms to compare some of the pictures. In this sense our proposed system can be categorized as a multi-modal system.

One of the main targets of data mining research is pattern discovery. Pattern discovery algorithms try to discover interesting patterns in a given database. Although LBP problems also focus on finding a pattern in a large amount of data, they cannot be classified as traditional pattern discovery problems. The imperfect tests (the biometric comparison algorithms) create noise in the system which does not allow us to use regular data-mining algorithms.

A melding of molecular biology with data-mining has created the field of bioinformatics. One of the main problems of the field is homology-search [2]. Two proteins are homologous if they have related folds and related sequences.

Popular homology search algorithms receive a new found protein as input, and search in protein databases for matches [9, 1]. Homology search shares several attributes with MLBP. Both problems search for target patterns in a large database. Both problems also have to positively classify sequences which are not a perfect match to the target pattern. However, while bioinformatics problems use static databases, the MLBP database rapidly changes. Popular sequence matching algorithms in bioinformatics [9, 1] assume cached databases in which extensive preprocessing has been done.

Moreover, LBP problems are different in the sense that the items in the stream of data are created by using a certain set of known rules. For example, if we know for certain that a pair of pictures does not belong to the same individual, we can prune any set of pictures which contain these images. This knowledge is exploited by the MLBP framework. Each pruned partial path saves processing of an exponential number of paths later.

We cannot use known classification algorithms such as Bayesian networks [6] or decision trees [4] to solve LBP problems in a centric way, using one computer. Without incremental pruning the system will fail to operate in real-time. One may suggest keeping the MLBP framework and the incremental pruning, but replacing the classifying mechanism. We have experimented with classifiers such as Bayesian-networks [6] and decision trees [4] but they did not yield any improvements. Due to lack of space we do not describe these trials here. Note that any alternative classifier must be simple enough to provide real-time solutions.

5 Conclusions

In this paper we addressed the domain of large-scale biometric pattern-recognition. The goal in LBP problems is to single out, in real-time, any individual who acts in a suspicious pattern. The combination of multiple locations and streams of input, inaccuracy of biometric tests and real-time constraints make LBP problems hard to solve, and they are ideal candidates for MAS implementation.

In this paper, we have presented the MLBP - a multi-agent solution to 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. Theoretical analysis of the local-module has provided upper bounds of the algorithm global performance. Simulations results of the problems show that the accuracy of the system is high while the number of innocent people detained is kept low. All results were obtained in real-time.

Future work can extend in three directions. First, we would like to solve new LBP problems using the MLBP framework. Second, we would like to generalize the algorithm to exploit more than two comparison algorithms. We would like to back the generalization with theoretical analysis. Intuitively, increasing the number of comparisons will improve overall performance. A third direction for future work is to include time performance into current theoretical analysis.

Current analysis can predict the false rejection and acceptance rates for a certain scenario. We would like to have a time estimation too.

References

1. S. F. Altschul, T. L. Madden, A. A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. Gapped BLAST and PSI–BLAST: a new generation of protein database search programs. *Nucleic Acids Res.*, 25:3389–3402, 1997.
2. A. Baxevanis and B. Ouellette. *Bioinformatics. A Practical Guide to the Analysis of Genes and Proteins.* Wiley-Interscience, 1998.
3. U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors. *Advances in Knowledge Discovery and Data Mining.* AAAI/MIT Press, Menlo Park, CA, 1996.
4. S. French. *Decision Theory - Introduction to the Mathematics of Rationality.* Ellis Horwood, London, 1993.
5. D. Hand, H. Mannila, and P. Smyth. *Principles of Data Mining.* MIT Press, 2001.
6. D. Heckerman, E. Horvitz, and B. Nathwani. Toward normative expert systems: Part I. The Pathfinder project. *SIAM J. Comput.*, 31:90–105, 1992.
7. L. Hong and A. K. Jain. Integrating faces and fingerprints for personal identification. *IEEE transactions PAMI*, 20(12):1295–1307, 1998.
8. A. K. Jain, A. Ross, and S. Prabhakar. Introduction to biometric recognition. *Transactions on Circuits and Systems for Video Technology*, 14(1), 2004.
9. W. Lipman, D.J. & Pearson. Rapid & sensitive protein similarity searches. *Science*, 227:1435–1441, 1985.
10. P. J. Phillips, P. Grother, R. J. Micheals, D. M. Blackburn, E. Tabassi, and J. M. Bone. FRVT 2002: Overview and summary. In *Proc. of Face Recognition Vendor Test 2002*, Virginia, 2002.
11. M. Przybocki and A. Martin. The NIST speaker recognition evaluation series, 2002.
12. A. Ross and A. K. Jain. Multimodal biometrics: An overview. In *Proc. of 12th European Signal Processing Conf.*, pages 1221–1224, 2004.
13. Y. Wang, T. Tan, and A. K. Jain. Combining face and iris biometrics for identity verification. *Audio and Video-based Biometric Person Authentication*, pages 805–813, 2003.
14. J. L. Wayman. Fundamentals of biometric authentication technologies. *INT J. of Imaging and Graphics*, 1(1):93–97, Jan. 2001.
15. P. Weinstein, H. V. Parunak, P. Chiusano, and S. Brueckner. Agents swarming in semantic spaces to corroborate hypotheses. In *Proc. of AAMAS04*, 2004.