Learning Graph Matching Substitution Weights based on a Linear Regression

Shaima Algabli and Francesc Serratosa
Universitat Rovira i Virgili,Tarragona, Spain
{shaima.ahmed@estudiants.urv.cat, francesc.serratosa}@urv.cat

UNIVERSITAT ROVIRA i VIRGILI

Introduction

Attributed graphs are structures that are useful to represent objects through the information of their local parts and their relations. Each characteristic in the local parts is represented by different attributes on the nodes. In this paper, we present **a method to learn the weights on each node attribute.**

Graph Edit Distance

The Graph Edit Distance :between two attributed graphs is defined as the transformation from one graph into another, through the edit operations, which obtains the minimum cost. These edit operations are: Substitution, deletion and insertion of nodes and also edges. Every edit operation has a cost depending on the attributes on the involved nodes or edges. This graph transformation can be defined through a node-to-node mapping f between nodes of both graphs.

$$ED(G,G') = \min_{\forall(p_1,\dots p_k)\in P(G,G')} \left\{ \underset{(p_1,\dots p_k)}{CED(G,G')} \right\}$$

where CED is the cost of the edit path,

$$\underset{(p_1,\dots p_k)}{CED(G,G')} = \sum_{\forall p_t \in vs} C_{vs}(p_t) + \sum_{\forall p_t \in es} C_{es}(p_t) + \sum_{\forall p_t \in vd} C_{vd}(p_t) + \sum_{\forall p_t \in ed} C_{ed}(p_t) + \sum_{\forall p_t \in vi} C_{vi}(p_t) + \sum_{\forall p_t \in ei} C_{ei}(p_t)$$

The method we present needs the substitution costs to be defined as a weighted Euclidean distance. Thus, if p_t is a node substitution that substitutes G^a by G'^i then $C_{vs}(p_t) = \sum_{k=1..K_v} w_{vs(k)} \cdot \left| v_k^a - v'^i_k \right|$

Learning model

The aim of the method is to learn $w = (w_1, \dots w_K)$Our learning method is based on two steps:

A. *Embedding the node-to-node mappings :*embeds the ground truth node-to-node mappings into a Euclidean space $S = (S_1, \dots, S_K)$.

B. *The learning algorithm:*deduces the linear regression of the embedded points in S to deduce the weights $w = (w_1, \dots w_K)$

The learning algorithm learns the weights w by finding the regression hyper-plane $w_1 \cdot S_1^a +,,, +w_K \cdot S_K^a = 0$, Our learnTing algorithm is composed of the following steps.First, M is computed,

$$M = \begin{bmatrix} S_1^{1^1} & \dots & S_k^{1^1} & \dots & S_K^{1^1} \\ \dots & & \dots & & \\ S_1^{N^{1^1}} & \dots & S_k^{N^{1^1}} & \dots & S_K^{N^{1^1}} \\ & & & & \\ & \dots & \dots & \dots & \\ & & & & \\ S_1^{1^P} & \dots & S_k^{1^P} & \dots & S_K^{N^{1^P}} \\ \dots & & & \dots & \\ S_1^{N^{P^P}} & \dots & S_k^{N^{P^P}} & \dots & S_K^{N^{P^P}} \end{bmatrix}$$

The computational cost of deducing $S_k^{a^r}$ is linear with respect to the number of node-to-node mappings in register r, N^r. Then the computational cost of generating M approximately is $O(K \cdot P \cdot N^2)$, being K the number of attributes, P the number of registers and N the number of nodes per graph. This is the highest cost given the three steps of our algorithm

- Second, the symmetric matrix $\Sigma = M^T \cdot$ M is computed.
- Third, the eigenvectors and eigenvalues of Σ are computed.
- And finally, weights $w = (w_1, \dots w_K)$ are defined as the eigenvector that has the minimum eigenvalue (Lagrange multipliers).

		(x,y)		
	Noise	0.1	0.2	0.3
α	0.1	DB1	DB5	DB9
	0.2	DB2	DB6	DB10
	0.3	DB3	DB7	DB11
	0.4	DB4	DB8	DB12

Table 1. The twelve used databases ordered by the level of noise on the position and the angle of the minutia.

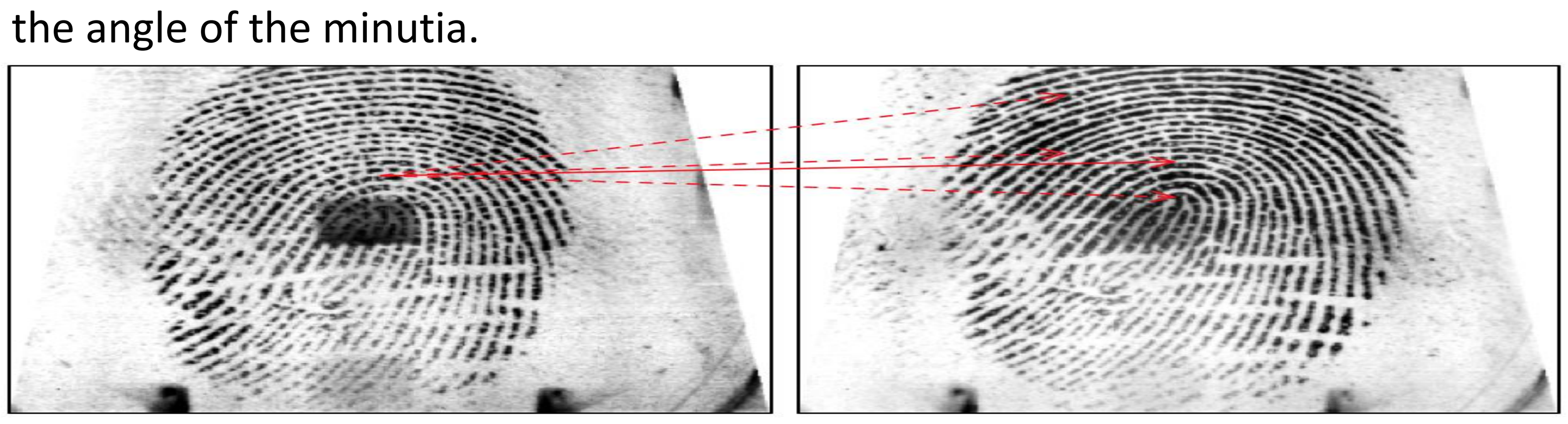

Fig. 1. Continuous arrow: Ground-truth minutia mapping. Dashed arrows: non ground-truth minutia mappings.

Experimental evaluation

- Fingerprint verification have been selected as the test application. Note that other applications could be used, such as hand-written character recognition based on graphs or scene interpretation, between others. We have selected fingerprint verification since the attributes on nodes represent completely different types of information (the 2D position and the angle).

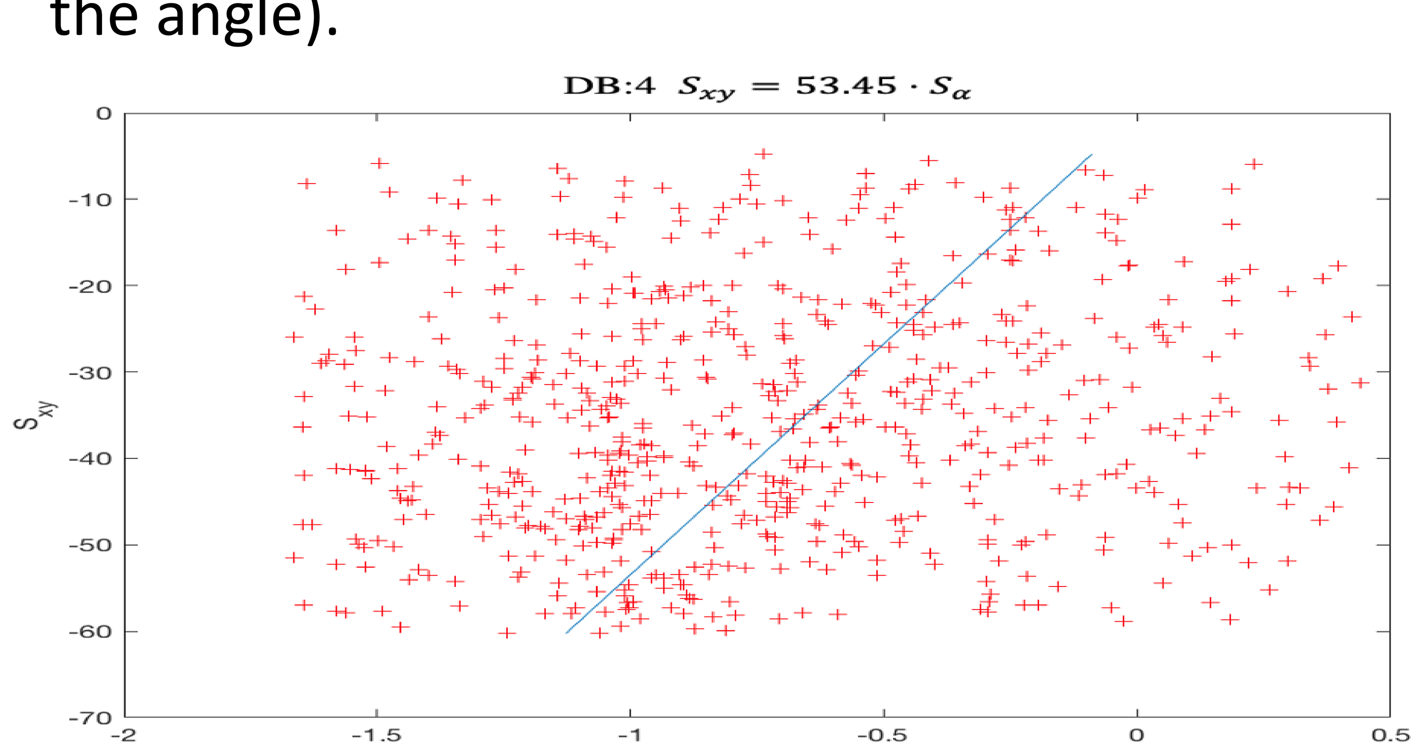

Fig. 2. Embedding space of DB4 and the deduced liner equation $S_{xy} = -\frac{w_{xy}}{w_\alpha} S_\alpha$.

Conclusions

Method	DB	1	2	3	4	5	6	7	8	9	10	11	12
Our method 0.05sec	w_α	30	39	43	53	36	40	40	42	35	38	47	56
	CR	1	1	0.98	0.91	1	1	0.9	0.8	1	0.91	0.78	0.56
	H	0.01	0.05	0.06	0.19	0.07	0.10	0.15	0.20	0.07	0.12	0.19	0.25
[15] 10 min	w_α	28	39	46	52	38	40	40	44	29	37	45	63
	CR	.9	1	0.95	0.90	1	0.97	0.89	0.81	1	0.92	0.81	0.57
	H	0.03	0.15	0.16	0.22	0.09	0.11	0.12	0.18	0.07	0.2	0.18	0.23
[16] 5 min	w_α	38	24	10	4	250	142	37	42	406	72	55	59
	CR	1	1	1	1	1	0.92	0.91	0.87	1	0.93	0.77	0.57
	H	0.01	0.04	0.03	0.08	0.08	0.14	0.15	0.18	0.06	0.12	0.19	0.25
[18] 8 min	w_α	30	39	Inf	Inf	36	41	Inf	42	-	-	48	64
	CR	1	1	0.12	0.03	1	1	0.01	0.8	-	-	0.78	0.55
	H	0.01	0.05	0.66	0.67	0.07	0.10	0.63	0.20	-	-	0.19	0.26

Table 2. Classification Ratio (CR) and Hamming distance (H) between the ground-truth mappings and=the deduced mappings given our method and methods [15,16,18] tested in 12 databases. It is also shown the deduced slope of the line, -$wxyw\alpha$. Moreover, the average learning runtime is shown in the first column.

This research is supported by the Spanish projects TIN2016-77836-C2-1-R and DPI2016-78957-R.