Distinguishing surgical behavior by sequential pattern discovery

Arnaud Huaulmé, Sandrine Voros, Laurent Riffaud, Germain Forestier, Alexandre Moreau-Gaudry, Pierre Jannin

Distinguishing Surgical Behavior by Sequential Pattern Discovery

Arnaud Huaulmé[a,b,c,*], Sandrine Voros[a], Laurent Riffaud[b,c,d], Germain Forestier[e], Alexandre Moreau-Gaudry[a,f], Pierre Jannin[b,c]

[a] *UGA / CNRS / INSERM, TIMC-IMAG UMR 5525, Grenoble, F-38041, France*
[b] *INSERM, UMR 1099, Rennes, F-35000, France*
[c] *Université de Rennes 1, LTSI, Rennes, F-35000, Franc*
[d] *Department of Neurosurgery, Rennes University Hospital,35000 Rennes, France*
[e] *MIPS, University of Haute-Alsace, France*
[f] *UGA / CHU Grenoble / INSERM, Centre d'Investigation Clinique - Innovation Technologique, CIT803, Grenoble, F-38041, France*

Abstract

Objective: Each surgical procedure is unique due to patient's and also surgeon's particularities. In this study, we propose a new approach to distinguish surgical behaviors between surgical sites, levels of expertise and individual surgeons thanks to a pattern discovery method. Methods: The developed approach aims to distinguish surgical behaviors based on shared longest frequent sequential patterns between surgical process models. To allow clustering, we propose a new metric called SLFSP. The approach is validated by comparison with a clustering method using Dynamic Time Warping as a metric to characterize the similarity between surgical process models. Results: Our method outperformed the existing approach. It was able to make a perfect distinction between surgical sites (accuracy of 100%). We reached an accuracy superior to 90% and 85% for distinguishing levels of expertise and individual surgeons. Conclusion: Clustering based on shared longest frequent sequential patterns outperformed the previous study based on time analysis. Significance: The proposed method shows the feasibility of comparing surgical process models, not only by their duration but also by their structure of activities. Furthermore, patterns may show risky behaviors, which could be an interesting information for surgical training to prevent adverse events.

Keywords: Pattern Discovery, Surgical Procedure, Surgical Process Model, Surgical Skills

1. Introduction

Surgical procedures are unique due to a patient's anatomical particularities, and due to the habits and experience of the surgical team. The surgical process modeling methodology was introduced about 15 years ago to study such variabilities [1, 2]. A surgical process model describes a surgical procedure at different levels of granularity as a list of phases, steps, and/or activities [2]. A surgical procedure can be divided into successive phases corresponding to the main periods of the procedure. A phase is composed of one or several steps. A step is a sequence of activities used to achieve a surgical objective. An activity is a physical action performed by the surgeon. Each activity is decomposed into different components, including the action verb, the target concerned by the action (anatomical structure), and the surgical instrument used to perform this action. Surgical Process Models (SPM) have been developed for three main purposes: (1) Formalizing surgical knowledge, (2) evaluating surgical skills and systems, (3) assisting the surgeon during a surgical intervention.

A SPM can be acquired manually from observations [3] or automatically thanks to recent advances in the automatic recognition of phases [4, 5], steps [6, 7] or activities [8, 9]. These SPMs have been recently used to identify differences between surgical behaviors, like the differences between surgical sites [10], surgical skills [11], types of procedures [12] and levels of surgical expertise [10, 13, 14].

In these studies, the differences between surgical behaviors were mostly computed by comparison of the surgical duration [10, 11, 13, 14]. For instance, Riffaud *et al.* [11] highlighted statistical relationship between surgical practice and surgical experience based on duration. Some studies computed a generic SPM (gSPM) to express the differences between surgical behaviors. For instance, Neumuth *et al.* [12] compared inpatient and outpatient procedures by creating one gSPM for each type of procedure, and used duration as metric to highlight the differences. However, a surgical behavior is not only characterized by the duration, but also by the succession of activities. To allow a better identification of surgical behavior, we present a method based on pattern discovery to identify a sequence of activities specific to a population of surgical cases.

Pattern discovery is used in various domains, like biology [15, 16], telecommunications [17], web [18] or medicine [19, 20, 21]. In all applications, the authors try to identify patterns (i.e. successions of elements that have multiple occurrences in a given set of sequences) to explain or understand the apparition of phenomena, for example, the apparition of an alert signal in telecommunications, or a protein promoter site in biology. In the medical field, the patterns are used to create a generic representation of the clinical pathway of patients from the admission to the discharge [19, 20] or to identify relationships between

*Corresponding author
Email address: arnaud.huaulme@imag.fr (Arnaud Huaulmé)

peri-operative data [21]. Up to our best knowledge, pattern discovery has never been used to identify surgical behaviors yet.

A pattern uniquely specific to a surgeon is a mark of his (her) behavior. We developed a pattern discovery method exploiting this characteristic based on expansion from short patterns, similar to the APRIORI method [22], for identifying differences in surgical behaviors.

2. Method

The aim of this study is to find the longest frequent patterns in surgical procedures and cluster these procedures thanks to the amount of patterns shared between them. For this purpose, we developed a pattern discovery method (section 2.1), and defined a new similarity metric for the clustering of surgical procedures (section 2.2).

2.1. Sequential Pattern Discovery

Sequential pattern discovery consists in finding patterns in sequences, where each element is directly followed by another. In our case, the sequences are surgical process models (SPMs) at the granularity level of activities performed by the surgeon's dominant hand. Therefore, a sequential pattern is a series of activities, at least 2, where an activity is followed by another determined activity. In our method, we search for longest frequent patterns. A pattern is frequent if the succession of activities it represents is present at least min_fr times in a set of SPMs, where min_fr is a predetermined threshold. We search the longest pattern in agreement with the lemma 1 of Mannila *et al.* [17]: "If an episode α is frequent in an event sequence s, then all subepisodes $\beta \preceq \alpha$ are frequent", where an episode is a pattern. Therefore, with longest patterns, we limit the number of results without any loss of information. The following subsections present the developed algorithms.

2.1.1. Main algorithm (algorithm 1)

Algorithm 1 computes all the longest frequent patterns present in a set of activity sequences. In a first step, the algorithm establishes a vocabulary of frequent activities (algorithm 2). Secondly, the algorithm generates candidate patterns of size k from the frequent patterns of size k-1 and frequent activities (algorithm 3). In a third step, the frequency of candidate patterns is computed to determine the frequent patterns of size k, and the longest frequent patterns of size k-1 (algorithm 4). Steps 2 and 3 are repeated to extend patterns until we do not find any new frequent patterns of size k. At each loop, the longest frequent patterns of size k-1 are added to the longest frequent patterns of inferior sizes. Figure 1 illustrates the full process for a simple example.

2.1.2. Get Vocabulary Activities algorithm (algorithm 2)

This algorithm extracts a vocabulary of frequent activities. First, it parses each sequence *s* (line 4) to count the number of apparitions of each activity. Them (line 13), if the number of apparitions is superior to the threshold min_fr, the activity is added to the frequent activities vocabulary.

```
   input : S: A set of activity sequences. min_fr: A
           frequency threshold.
   output: allLongestFrequentPatterns: A set of longest
           frequent patterns.
 1 begin
 2     frequentActivities ← Get Vocabulary
       Activities(S, min_fr);     /* Algorithm 2 */
 3     k = 2;                     /* Length of patterns */
 4     frequentPatterns_{k-1} ←frequentActivities;
 5     while |frequentPatterns_{k-1} | > 0 do
 6         candidatePatterns ← Get Candidate
           Patterns(frequentPatterns_{k-1},
           frequentActivities, k);     /* Algorithm 3 */
 7         frequentPatterns_k,allLongestFrequentPatterns
           ← Get Frequent Patterns(min_fr, k, S,
           candidatePatterns);     /* Algorithm 4 */
 8         k++;
 9         frequentPatterns_{k-1} ←frequentPatterns_k;
10     end
11     return allLongestFrequentPatterns;
12 end
```

Algorithm 1: Main Algorithm.

```
 1 Get Vocabulary Activities(S, min_fr)
   input : S: A set of activity sequences. min_fr: A
           frequency threshold.
   output: frequentActivities: A set of frequent activities.
 2 begin
 3     /* Count frequency of apparition of each
          activity.                                  */
 4     foreach sequence s of S do
 5         for i=1 to |s| do
 6             if s[i]∉activitiesMap then
 7                 activitiesMap.insert(s[i],1) ;
                   /* activitiesMap contains
                   activities (keys) and their
                   frequency of apparition
                   (values) */
 8             end
 9             else activitiesMap [s[i]]++ ;
10         end
11     end
12     /* Save frequent activities in
          frequentActivities.                        */
13     foreach activity a of activitiesMap do
14         if a.value ≥min_fr then
15             frequentActivities.insert(a.key);
16         end
17     end
18     return frequentActivities;
19 end
```

Algorithm 2: Algorithm to return all frequent ($>min\_fr$) activities in a set of sequences S.

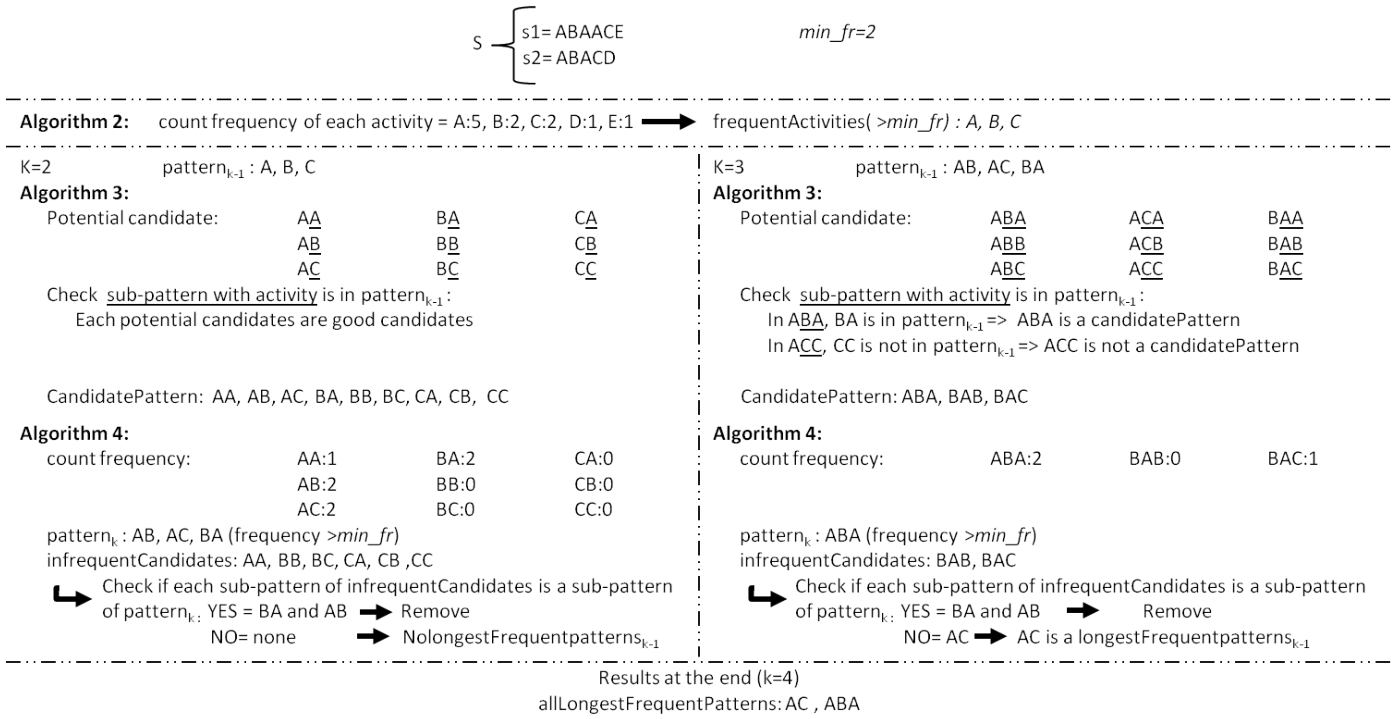

Figure 1: Proceeding of the sequential pattern discovery method for a simple example. S is a set of activity sequences, min_fr is a frequency threshold.

With this vocabulary of frequent activities, we are certain that all the patterns constructed with these activities can be frequent. The vocabulary of frequent activities limits the candidate patterns created in algorithm 3.

2.1.3. Get Candidate Patterns algorithm (algorithm 3)

The aim of this algorithm is to return candidate patterns that could be frequent (good candidate).

First, an activity a of the vocabulary of frequent activities is added at the end of a frequent pattern of size k-1, to create a potential candidate of size k (line 5). At this instant, every activity constituing the potential candidate is frequent (thanks to algorithm 2), but we do not know if each sub-pattern is frequent. Therefore, we test if the sub-patterns of size k-1 are frequent (line 6). We test the sub-pattern containing the activity a only, since the other, due to the construction of a potential candidate, is frequent. If the sub-pattern containing the activity a is one of the frequent patterns of size k-1, the potential candidate is a good candidate. This is repeated for each frequent pattern of size k-1 associated to each activity.

2.1.4. Get Frequent Patterns algorithm (algorithm 4)

This algorithm has two functions: (1) return the frequent patterns of size k, and (2) return the longest frequent patterns of size k-1.

First, the algorithm counts the number of apparitions of each candidate pattern in activity sequences S (lines 4 to 13). The number of candidate patterns can be important, especially for candidate patterns of 2 activities: Indeed, for 100 frequent activities, algorithm 3 gives 10,000 candidate patterns (100 x 100). To improve the computation time, instead of using brute

```
1  Get Candidate Patterns(patterns_{k-1}, frequentActivities, k)
   input : patterns_{k-1}: A set of frequent patterns of k-1 activities. frequentActivities: A set of frequent activities. k: The length of patterns to generate.
   output: candidatePatterns: A set of candidate patterns of length k.
2  begin
3      for i=1 to |patterns_{k-1}| do
4          foreach activity a of frequentActivities do
5              potentialCandidate ← Concatenation(patterns_{k-1}[i], a);
               /* Add activity a at the end of the i^th pattern of patterns_{k-1} */
6              if potentialCandidate [2,k] ∈ patterns_{k-1} then
7                  /* if sub-pattern of size k-1 with a is frequent */
8                  candidatePatterns.insert (potentialCandidate);
9              end
10         end
11     end
12     return candidatePatterns;
13 end
```

Algorithm 3: Algorithm to generate candidates of length k thanks to the frequent patterns of length k-1 ($patterns_{k-1}$) and the frequent activities.

```
1  Get Frequent Patterns(min_fr, k, S, candidatePatterns)
   input  : min_fr: A frequency threshold. k: The length of patterns. S: A set of activity sequences. candidatePatterns: A set
            of candidate patterns of length k.
   output: patterns_k: A set of frequent pattern of k activities. longestFrequentpatterns_{k-1}: A set of frequent pattern of k-1
            activities not present in patterns_k.
2  begin
3      /* Count frequency of apparition of each candidate c of candidatePatterns in S                    */
4      foreach sequence s of S do
5          for start=1 to |s|−k do
6              foreach candidate c of candidatePatterns do
7                  if s[start, start+k ]=c then
8                      c.freq++;
9                      start++; Stop and go to line 5
10                 end
11             end
12         end
13     end
14     foreach candidate c of candidatePatterns do
15         /* save frequent candidates in patterns_k                                                     */
16         if c.freq ≥ min_fr then patterns_k.insert(c);
17         else infrequentCandidates.insert(c);
18     end
19     /* Save sub-patterns of infrequentCandidates were not sub-pattern of patterns_k                  */
20     foreach infrequent candidate i of infrequentCandidates do
21         for start=1 to 2 do
22             subPattern←i[start, k-2+start];
23             if !Contains Sub-pattern(patterns_k, subPattern) then
24                 longestFrequentpatterns_{k-1}.insert(subPattern);
25             end
26         end
27     end
28     return patterns_k, longestFrequentpatterns_{k-1};
29 end
```

Algorithm 4: Algorithm to get a set of frequent pattern of k activities ($patterns_k$) and a set of longest frequent pattern of k-1 activities ($longestFrequentpatterns_{k-1}$).

search, we look through each activity sequence only once, and compare a window of size k to each candidate c (line 7). If a sequence window matches a candidate pattern c, the number of apparitions of c is incremented (line 8), we do not test the remaining candidates and we shift the position of the windows (line 9).

The second step (lines 14 to 18) consists in checking if the number of apparitions of candidate c is superior to the threshold min_fr. This step separates candidate patterns into frequent patterns ($patterns_k$) and infrequent candidates ($infrequentCandidates$). The frequent patterns will be used in the next loop of algorithm 1.

The infrequent candidates are composed of frequent sub-patterns (algorithm 3). The last step (lines 20 to 27 of algorithm 4) consists in checking if these sub-patterns are contained in the frequent patterns ($patterns_k$). If it is not the case, these frequent sub-patterns constitute the longest frequent patterns of size k-1.

2.2. Hierarchical Clustering

Clustering [23] consists in the creation of sets of similar objects (called clusters). Hierarchical clustering is a bottom-up clustering method. First, each object constitutes its own cluster. And, clusters are iteratively merged by pairs into a higher-level cluster. To performed clustering, we need to specify a metric characterizing the similarity between surgical process models and an approach to merge the clusters. We propose a new metric called Shared Longest Frequent Sequential Pattern metric (SLFSP metric) based on the number of shared longest frequent sequential patterns between 2 SPMs divided by the number of unique longest frequent patterns of both SPMs:

$$SLFSP(A,B) = \frac{|shared_{A,B}|}{|patterns_A| + |patterns_B| - |shared_{A,B}|} \quad , \quad (1)$$

where A and B are 2 SPMs, $|shared_{A,B}|$ is the number of shared longest frequent sequential patterns between A and B, and $|patterns_A|$ and $|patterns_B|$ are respectively the number of longest frequent patterns of A and B.

We used the average-link approach with UPGMA algorithm (Unweighted Pair Group Method with Arithmetic Mean) [24] to merge the clusters. The average-link approach consists in evaluating the similarity between two clusters according to the average distance between all couples of objects in the two clusters. Therefore, the distance between clusters C_1 and C_2, composed of SPMs, is defined as:

$$d(C_1,C_2) = \frac{1}{|C_1||C_2|} \sum_{i=1}^{|C_1|} \sum_{j=1}^{|C_2|} SLFSP(spm_i, spm_j) \quad , \quad (2)$$

where $|C_n|$ is the number of SPMs in the cluster n.

To analyze the clusters, created by the hierarchical clustering, we use a dendrogram.

3. Validation

To validate our method, we compared our approach with the study of Forestier *et al.* [10]. In this study, the authors compared surgical practices between three different surgical sites through a Dynamic Time Warping distance. First, we briefly present the data. Secondly, we present the previous results obtained by the authors. Finally, we present the results with our method.

3.1. Data

The data consist of 41 one-level Anterior Cervical Discectomies (ACDs) performed in three different surgical sites by 11 different surgeons. During an ACD, a cervical disc is removed by an anterior approach. In the 41 surgeries, 11 were performed on site A, 12 on site B, and 18 on cite C (sites are anonymized). Two expertise levels were defined by Forestier *et al.* [10]: expert and intermediate. An expert is a neurosurgeon who has performed more than 200 ACDs, whereas an intermediate is a neurosurgeon who has performed less than 100 ACDs. Table 1 presents the number of surgeries, the number of experts surgeons and intermediates surgeons by surgical site.

Surgical site	A	B	C
Number of surgeries	11	12	18
Number of experts	3	3	2
Number of intermediates	1	0	2

Table 1: Repartition of surgeries and surgeon expertise by surgical sites.

The data were recorded on-line by two surgeons, an expert surgeon for site A and C, and an intermediate surgeon for site B. Both operators used the Surgical Workflow Editor [3] for data recording and possessed the same level of training in this editor. However, Forestier *et al.* [10] have shown a vocabulary heterogeneity between surgical sites, specifically, between site B and the others two (less than 50% of similarity). To remove this issue, the terms used on sites A and C were matched with the terms used on site B by an expert surgeon. For our study, we used the SPMs obtained after vocabulary uniformization.

3.2. Previous study

In their study, Forestier *et al.* [10] used Dynamic Time Warping distance as the metric to characterize similarity between surgical process models. This metric and an average-link approach are used to perform the hierarchical clustering.

The authors made three levels of analyses: distinguishing surgical sites, distinguishing individual surgeons and distinguishing levels of expertise. They had an accuracy of 97.5% (40/41) for distinguishing surgical sites, 72.4% (21/29) for distinguishing individual surgeons, and 86.2% (25/29) for distinguishing levels of expertise. The two last analyses were only performed for site A and C, because for surgical site B all surgeons were expert and no clear sub-clusters emerged .

The authors announced few minutes to compute the distance matrix.

3.3. Results

Figure 2 represents the results of our clustering method by pattern discovery with a threshold min_{fr} of 6. Which was selected because it is the threshold which offers the best results for clinical sites clustering. We identified 207 longest frequent patterns in the 41 surgeries. The computation time to find the patterns and to create the distance matrix was inferior to 30 seconds.

We identified 3 clusters (CL^A,CL^B, and CL^C), one for each surgical site. In reality CL^C is not a cluster, but 3 separate clusters with the same surgical site. Therefore, our method has an accuracy of 100% to identify surgical sites.

We separated each of the previous clusters into sub-clusters where the surgeries were performed by the same surgeon. These clusters are noted CL^n, where n is the id of the surgeon in this cluster. Only three surgeons have misclassified surgeries: (1) the two surgeries of surgeon 10 are classified with surgeon 9 (CL^9), (2) one surgery of surgeon 3 is classified in CL^4, and (3) surgeon 6 has one surgery in CL^5, one in CL^7 and one alone. Our method has an accuracy of 85.4% (35/41) for all surgical sites, and 89.6% (26/29) for surgical site A and C.

We also studied the clustering by levels of expertise. Like Forestier *et al.* [10], we did not perform this study for surgical site B due to the fact that all surgeons are experts. Only CL^9 has different level of expertise due to the presence of surgeries of the intermediate surgeon 10 in this expert cluster. Therefore, the accuracy we obtained for distinguishing the levels of expertise is 93.1% (27/29). These results are compared with those of Forestier *et al.* [10] in table 2.

4. Discussion

In this paper, we proposed an approach for studying surgical procedures based on pattern discovery. We applied this

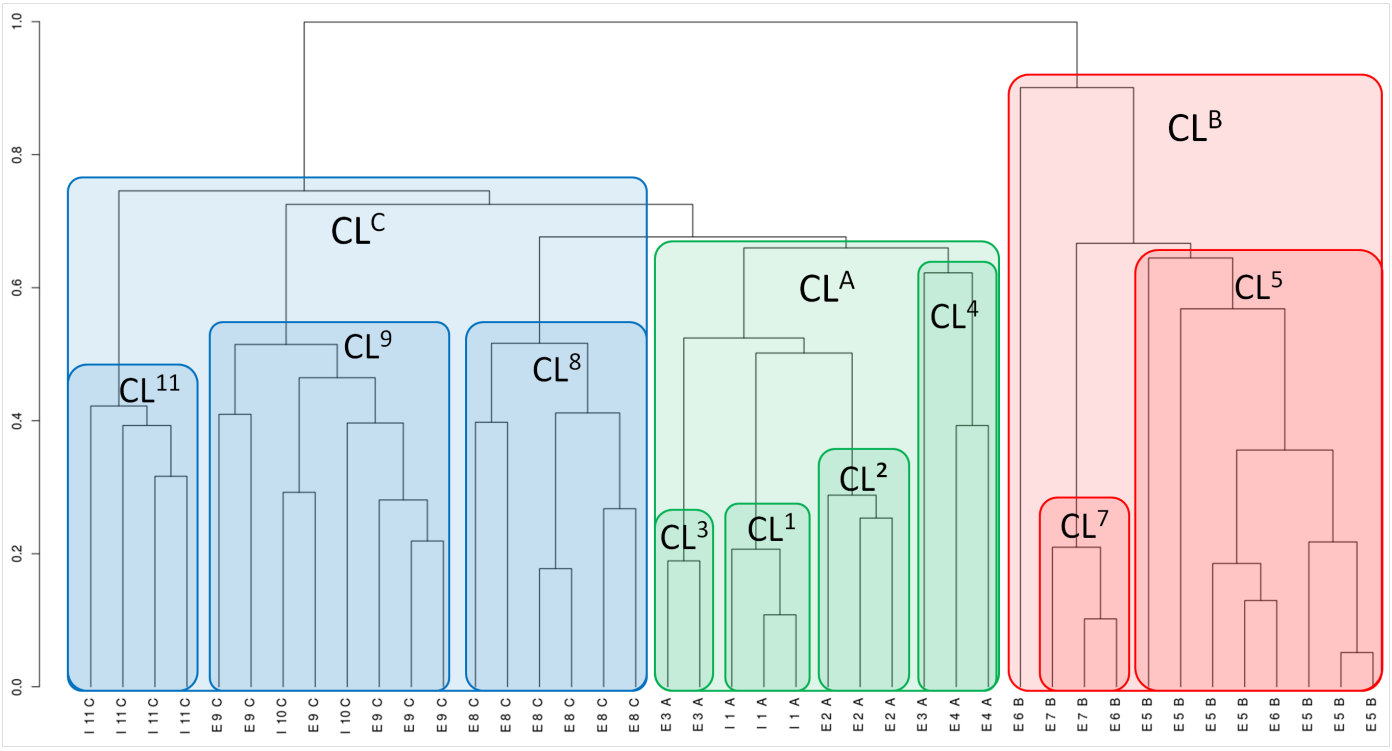

Figure 2: Dendrogram representing the hierarchical clustering based on our method, using 41 surgeries, for a threshold of 6. The horizontal axis represents the surgeries. The vertical axis represents the dissimilarity (1- SLFSP). For each surgery, the site (A, B, and C), the surgeon id (1-11) and the level of expertise (Expert (E), Intermediate (I)) are mentioned.

Analysis	Surgical sites	Levels of expertise	Individual surgeons (sites A and C)	Individual surgeons (all sites)
Accuracy for Forestier *et al.* study[10] (%)	97.5	86.2	72.4	Na
Accuracy for our method (%)	100	93.1	89.6	85.4

Table 2: Accuracy of distinguishing surgical sites, of levels of expertise and individual surgeons by both methods. Study of level of expertise was only performed for surgical site A and C. Na: Not applicable

approach to define a new metric called SLFSP. We compared our approach with an existing approach and demonstrated that pattern analysis had a better discriminative power.

Our method distinguishes the surgical sites with an accuracy of 100%, and clusters levels of expertise (only for surgical site A and C) with an accuracy of 93.1%. In these two cases, for the same number of surgical process models, we outperformed the results of Forestier *et al.* [10] (97.5% for clustering by surgical sites and 86.2% for clustering by level of expertise). Moreover, Forestier *et al.* proposed a clustering by surgeons only for site A and C, because no clear sub-clusters emerged with their method for site B, with an accuracy of 72.4%. In our case, for all surgical sites, we had an accuracy of 85.4%. Our method thus results in better clustering.

However, the accuracy for the surgical site classification must be nuanced. In figure 2, we can see that the cluster of the surgeon 8 (CL^8) has a distance closer to site A than other clusters of surgical site C (CL^{10} and CL^{11}). This result could be explained by the fact that surgical behavior is not completely different depending on the surgical site. However, the clear clusters obtained for sites A and C suggest that it does have an influence that could be explained by knowledge and practice sharing between surgeons working at the same surgical site.

Regarding the clustering by individual surgeons and by levels of expertise, CL^9 clusters an expert surgeon (9) and an intermediate surgeon (10) together. This could be explained by the fact that the surgeon 9 was the mentor of the surgeon 10, implying that they both were performing similar patterns.

The main limitation of our method is the necessity to define a threshold for the number of apparitions of the pattern, in order to limit the number of irrelevant patterns. In our case, we chose a threshold of 6 because it offers the best results for clinical sites clustering of our 41 surgeries. This threshold is dependent on the number of sequences, consequently, it is necessary to determine the best threshold by testing. However, it is possible to search for the optimal threshold by exhaustive search, due to the low computation time (less than 30 second with our threshold).

Even for an off-line method, it seems necessary that the cal-

culation time is as small as possible for a restricted data set, in order to be applicable to larger data set. Our method better addresses this issue, with a computation time inferior to 30 seconds, than Forestier *et al.*, where it is a few minutes.

5. Conclusion

With our clustering method, thanks to longest frequent patterns, we obtain better results than Forestier *et al.* study [10] for the same data.

For the computation of the SLFSP metric, we only used longest frequent patterns. It is possible that sub-patterns also give information. It will be interesting to add information of sub-patterns to allow better clustering.

In this study, we focused on fixed consecutive patterns. Currently, if two activities only differ by a surgical instrument, we considered them as two different activities. But if both surgical instruments have the same function (for example, to cut an anatomical structure), it could be logical to consider both these activities as equivalent. So, enhancing the method by allowing substitution of similar activities inside a pattern, according to ontological rules, may lead to interesting improvements.

This study highlights the possibility to identify specific surgical behaviors of different populations of surgeons. The objective of this work is the identification of surgical behaviors which may be correlated to the apparition of intraoperative adverse events. Such identification of behaviors may help preventing their apparition or limiting their severity.

Additionally, our approach may improve automatic recognition of surgical phases or steps, by providing some additional a priori knowledge on expected surgical patterns in complementary to other existing approaches such as where hidden Markov models or Dynamic Time Warping [4, 5, 6, 7].

Acknowledgements

This work was partly supported by French state funds managed by the ANR within the Investissements d'Avenir program (Labex CAMI) under reference ANR-11-LABX-0004.

[1] P. Jannin, M. Raimbault, X. Morandi, B. Gibaud, Modeling Surgical Procedures for Multimodal Image-Guided Neurosurgery, in: W. J. Niessen, M. A. Viergever (Eds.), Medical Image Computing and Computer-Assisted Intervention MICCAI 2001, no. 2208 in Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2001, pp. 565–572, dOI: 10.1007/3-540-45468-3_68.

[2] F. Lalys, P. Jannin, Surgical process modelling: a review, International Journal of Computer Assisted Radiology and Surgery 9 (3) (2013) 495–511. doi:10.1007/s11548-013-0940-5.

[3] T. Neumuth, G. Strau, J. Meixensberger, H. U. Lemke, O. Burgert, Acquisition of Process Descriptions from Surgical Interventions, in: S. Bressan, J. Kng, R. Wagner (Eds.), Database and Expert Systems Applications, no. 4080 in Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2006, pp. 602–611, dOI: 10.1007/11827405_59.

[4] N. Padoy, B. Tobias, H. Feussner, M.-O. Berger, N. Navab, On-line Recognition of Surgical Activity for Monitoring in the Operating Room, 2008, pp. 1718–1724.

[5] N. Padoy, T. Blum, S.-A. Ahmadi, H. Feussner, M.-O. Berger, N. Navab, Statistical modeling and recognition of surgical workflow, Medical Image Analysis 16 (3) (2010) 632–641. doi:10.1016/j.media.2010.10.001.

[6] L. Bouarfa, P. Jonker, J. Dankelman, Discovery of high-level tasks in the operating room, Journal of Biomedical Informatics 44 (3) (2011) 455–462. doi:10.1016/j.jbi.2010.01.004.

[7] A. James, D. Vieira, B. Lo, A. Darzi, G.-Z. Yang, Eye-Gaze Driven Surgical Workflow Segmentation, Medical Image Computing and Computer-Assisted Intervention MICCAI 2007 (2007) 110–117doi:10.1007/978-3-540-75759-7_14.

[8] S.-Y. Ko, J. Kim, W.-J. Lee, D.-S. Kwon, Surgery task model for intelligent interaction between surgeon and laparoscopic assistant robot, International Journal of Assitive Robotics and Mechatronics 8 (1) (2007) 38–46.

[9] F. Lalys, D. Bouget, L. Riffaud, P. Jannin, Automatic knowledge-based recognition of low-level tasks in ophthalmological procedures, International Journal of Computer Assisted Radiology and Surgery 8 (1) (2012) 39–49. doi:10.1007/s11548-012-0685-6.

[10] G. Forestier, F. Lalys, L. Riffaud, D. Louis Collins, J. Meixensberger, S. N. Wassef, T. Neumuth, B. Goulet, P. Jannin, Multi-site study of surgical practice in neurosurgery based on surgical process models, Journal of Biomedical Informatics 46 (5) (2013) 822–829. doi:10.1016/j.jbi.2013.06.006.

[11] L. Riffaud, T. Neumuth, X. Morandi, C. Trantakis, J. Meixensberger, O. Burgert, B. Trelhu, P. Jannin, Recording of Surgical Processes: A Study Comparing Senior and Junior Neurosurgeons During Lumbar Disc Herniation Surgery:, Operative Neurosurgery 67 (2010) ons325–ons332. doi:10.1227/NEU.0b013e3181f741d7.

[12] T. Neumuth, R. Wiedemann, C. Foja, P. Meier, J. Schlomberg, D. Neumuth, P. Wiedemann, Identification of surgeonindividual treatment profiles to support the provision of an optimum treatment service for cataract patients, Journal of Ocular Biology, Diseases, and Informatics 3 (2) (2010) 73–83. doi:10.1007/s12177-011-9058-6.

[13] G. Forestier, F. Lalys, L. Riffaud, B. Trelhu, P. Jannin, Classification of surgical processes using dynamic time warping, Journal of biomedical informatics 45 (2) (2012) 255–264.

[14] C. Cao, C. L. MacKenzie, S. Payandeh, Task and motion analyses in endoscopic surgery, 1996, pp. 583–590.

[15] D. Y. Chiang, P. O. Brown, M. B. Eisen, Visualizing associations between genome sequences and gene expression data using genome-mean expression profiles, Bioinformatics 17 (suppl 1) (2001) S49–S55.

[16] I. Eidhammer, I. Jonassen, W. R. Taylor, Structure Comparison and Structure Patterns, Journal of Computational Biology 7 (5) (2000) 685–716. doi:10.1089/106652701446152.

[17] H. Mannila, H. Toivonen, A. I. Verkamo, Discovery of Frequent Episodes in Event Sequences, Data Mining and Knowledge Discovery 1 (3) (1997) 259–289. doi:10.1023/A:1009748302351.

[18] C. I. Ezeife, Y. Lu, Mining Web Log Sequential Patterns with Position Coded Pre-Order Linked WAP-Tree, Data Mining and Knowledge Discovery 10 (1) (2005) 5–38. doi:10.1007/s10618-005-0248-3.

[19] Z. Huang, X. Lu, H. Duan, W. Fan, Summarizing clinical pathways from event logs, Journal of Biomedical Informatics 46 (1) (2013) 111–127. doi:10.1016/j.jbi.2012.10.001.

[20] Z. Huang, W. Dong, L. Ji, C. Gan, X. Lu, H. Duan, Discovery of clinical pathway patterns from event logs using probabilistic topic models, Journal of Biomedical Informatics 47 (2014) 39–57. doi:10.1016/j.jbi.2013.09.003.

[21] I. M. Mullins, M. S. Siadaty, J. Lyman, K. Scully, C. T. Garrett, W. Greg Miller, R. Muller, B. Robson, C. Apte, S. Weiss, I. Rigoutsos, D. Platt, S. Cohen, W. A. Knaus, Data mining and clinical data repositories: Insights from a 667,000 patient data set, Computers in Biology and Medicine 36 (12) (2006) 1351–1377. doi:10.1016/j.compbiomed.2005.08.003.

[22] R. Agrawal, R. Srikant, Fast Algorithms For Mining Association Rules In Datamining (1994) 487–499.

[23] A. K. Jain, Data clustering: 50 years beyond K-means, Pattern Recognition Letters 31 (8) (2010) 651–666. doi:10.1016/j.patrec.2009.09.011.

[24] R. R. Sokal, C. D. Michener, A statistical method for evaluating systematic relationships, University of Kansas Scientific Bulletin 28 (1958) 1409–1438.