Pattern discovery in annotated dialogues using dynamic programming

Émilie Chanoni, Thierry Lecroq, Alexandre Pauchet

Abstract—We describe a scenario to discover patterns in annotated dialogues: first dialogues are transcripted and annotated; then two-dimensional patterns are extracted; finally the semantic pertinence of the extracted patterns can be evaluated by an expert. The dialogues are annotated by using a grid designed by the expert. Recurrent patterns are extracted in the annotated dialogues using dynamic programming with the help of a substitution matrix specifically designed for the task. In this article we focus on the method developed for extracting the patterns and we show some extracted patterns on annotation of dialogues between parents and four year old children during the narration of child stories whose context is mentalist.

Index Terms—Dynamic programming, Data mining, Psychology

1 INTRODUCTION

The design of computer systems that interact efficiently with users necessitates robust models. These models have not to be defined arbitrary by the system designer but they have to be based on the study of human capabilities of interaction and communication. For example, a friendly dialogic interface in natural language has to be efficient concerning the task model and has to integrate a communication behavior as natural as possible. Among the various existing dialogue and interaction models, the most common approach adopted is to study a corpus of annotated dialogues, traces or logs, in order to extract a set of significant recurrent behaviors. When the annotations are judiciously chosen, the recurrent behaviors appear as patterns repeated in the annotations. These repetitions are often represented with automata [26], timed automata [19], Petri nets [16], sequence diagrams [24], and so on. All these representations use linear patterns.

In the particular case of corpus of dialogues whose annotation are aimed at characterizing a mentalist process, each utterance is annotated with a series of codes. Therefore, extracting two dimensional patterns in such annotated dialogues requires a approach different from the classical one.

The method presented in this article answers to this problematic thanks to a scenario during which expert knowledge is as important as the algorithm:

1) Transcription and annotation of the dialogues;
2) Extracting alignments of two-dimensional patterns using dynamic programming;
3) Evaluation of the semantic pertinence of the extracted patterns (expert knowledge);

Dynamic programming is a technique used in many applications (see [15], [11], [23], [22]). Strings (sequences of symbols belonging to an alphabet \(\Sigma\)) can easily be compared using dynamic programming. Applications of this problem include computational biology, computation linguistics or speech recognition. Trees can also be compared using similar techniques. The reader is referred to [20] for further details. There is a great number of works considering exact and fuzzy pattern (or approximate pattern) matching in two dimensions: given a pattern \(X\) and a text \(Y\), find the occurrences of \(X\) in \(Y\) (see [2], [12], [3]). Very little attention has been given to the alignments of two-dimensional matrices though applications of this problem include fundamental topics such as data mining (in time series databases for instance) and image analysis. Krithivasan and Sitalakshmi [14] consider 2D patterns of exactly the same size while Baeza-Yates [6] considers only global alignments of 2D patterns. There has been some efforts for indexing matrices using either suffix trees or suffix arrays (see [10], [13], [18]) but those indexes enable to find exact repeats and the generalization for finding approximate repeats is far from immediate. Recently, Arslan [5] considered the largest common subsequence of two patterns in dimension \(d\) but using different techniques. In [4], the authors proved that finding the longest common substrutures of two matrices in \(NP\)-hard.

We present here new heuristics for comparing two matrices, more precisely we give the recurrence formulas for performing global as well as local alignments of two-dimensional patterns of respective size \(M\) and \(N\) in...
Let us define the edit distance between two strings $x$ and $y$ as the minimum cost of elementary edit operations that enable to transform $x$ into $y$. The elementary edit operations are: the substitution of a symbol of $x$ at a given position by a symbol of $y$; the deletion of a symbol of $x$ at a given position; the insertion of a symbol of $y$ in $x$ at a given position.

A cost is associated to each elementary edit operation. For $a, b \in \Sigma$: $Sub(a, b)$ denotes the cost of the substitution of the symbol $a$ by the symbol $b$, $Del(a)$ denotes the cost of the deletion of the symbol $a$, $Ins(a)$ denotes the cost of the insertion of the symbol $a$. This means that the costs of the edit operations are independent of the positions where the operations occur. We can now define the edit distance of two strings $x$ and $y$ by:

$$d(x, y) = \min \{ \text{cost of } \gamma \mid \gamma \in \Gamma_{x,y} \}$$

where $\Gamma_{x,y}$ is the set of all the sequences of edit operations that transform $x$ into $y$, and the cost of an element $\gamma \in \Gamma_{x,y}$ is the sum of the costs of its elementary edit operations.

In order to compute $d(x, y)$ for two strings $x$ and $y$ of length $m$ and $n$ respectively, we use a two-dimensional table $t$ of $m+1$ rows and $n+1$ columns such that $t[i, j] = d(x[0 \ldots i], y[0 \ldots j])$ for $i = 0, \ldots, m-1$ and $j = 0, \ldots, n-1$. It follows $d(x, y) = t[m-1, n-1]$.

The values of the table $t$ can be computed by the following recurrence formulas for $i = 0, 1, \ldots, m-1$ and $j = 0, 1, \ldots, n-1$:

$$t[i-1, j] = 0$$

$$t[i, j-1] = t[i, j-1] + Del(x[i])$$

$$t[i, j] = \min \left\{ t[i-1, j-1] + Sub(x[i], y[j]), t[i-1, j] + Del(x[i]), t[i, j-1] + Ins(y[j]) \right\}$$

The value at position $(i, j)$ in the table $t$ only depends on the values at the three neighbor positions $(i-1, j-1)$, $(i-1, j)$ and $(i, j-1)$ (see [8]).

The direct application of the above recurrence formula gives an exponential time algorithm to compute $t[m-1, n-1]$. However the whole table $t$ can be computed in quadratic time, technique known as “dynamic programming”. This is a general technique that is used to solve the different kinds of alignments.

An optimal alignment (with minimal cost) can then be produced. It consists in tracing back the computation of the values of the table $t$ from position $(m-1, n-1)$ to position $(-1, -1)$. At each cell $[i, j]$ the algorithm determines among the three values $t[i-1, j-1] + Sub(x[i], y[j])$, $t[i-1, j] + Del(x[i])$ and $t[i, j-1] + Ins(y[j])$ which has been used to produce the value of $t[i, j]$. If $t[i-1, j-1] + Sub(x[i], y[j])$ has been used it adds $(x[i], y[j])$ to the optimal alignment and proceeds recursively with cell at $[i-1, j-1]$. If $t[i-1, j] + Del(x[i])$ has been used it adds $(x[i], -)$ to the optimal alignment and proceeds recursively with cell at $[i-1, j]$. If $t[i, j-1] + Ins(y[j])$ has been used it adds $(-, y[j])$ to the optimal alignment and proceeds recursively with cell at $[i, j-1]$.
proceeds recursively with cell at \([i, j - 1]\). Recovering all the optimal alignments can be done by a similar technique.

### 2.2 Local alignment

A local alignment of two strings \(x\) and \(y\) consists in finding the segment of \(x\) that is closer to a segment of \(y\). The notion of distance used to compute global alignments cannot be used in that case since the segments of \(x\) closer to segments of \(y\) would only be the empty segment or individual symbols. This is why a notion of similarity is used based on a scoring scheme for edit operations (see [21]).

A score (instead of a cost) is associated to each elementary edit operation. For \(a, b \in \Sigma\): \(\text{Sub}_S(a, b)\) denotes the score of substituting the symbol \(b\) for the symbol \(a\), \(\text{Del}_S(a)\) denotes the score of deleting the symbol \(a\), \(\text{Ins}_S(a)\) denotes the score of inserting the symbol \(a\). For two symbols \(a\) and \(b\), a positive value of \(\text{Sub}_S(a, b)\) means that the two symbols are close to each other, and a negative value of \(\text{Sub}_S(a, b)\) means that the two symbols are far apart.

We can now define the edit score of two strings \(x\) and \(y\) by \(s(x, y)\) = maximal similarity between a segment of \(x\) and a segment of \(y\).

In order to compute \(s(x, y)\) for two strings \(x\) and \(y\) of length \(m\) and \(n\) respectively, we make use of a two-dimensional table \(t_S\) of \(m + 1\) rows and \(n + 1\) columns such that \(t_S[i, j] = \max\{s(x[\ell \ldots i], y[k \ldots j]) | 0 \leq \ell \leq i \text{ and } 0 \leq k \leq j \} \cup \{0\}\), for \(i = 0, \ldots, m - 1\) and \(j = 0, \ldots, n - 1\). Therefore, \(s(x, y)\) = maximal value in \(t_S\).

The values of the table \(t_S\) can be computed by the following recurrence formula for \(i = 0, 1, \ldots, m - 1\) and \(j = 0, 1, \ldots, n - 1\):

\[
\begin{align*}
t_S[-1, -1] &= t_S[-1, 1] = t_S[-1, j] = 0, \\
\begin{cases} t_S[-1, 1] + \text{Sub}_S(x[i], y[j]), \\
\text{Del}_S(x[i]), \\
\text{Ins}_S(y[j]), \\
\end{cases} &\quad i, j = 0, 1, \ldots, n - 1,
\end{align*}
\]

Computing the values of \(t_S\) for a local alignment of \(x\) and \(y\) can be done in \(O(mn)\) time and space complexity. Recovering a local alignment can be done in a way similar to what is done in the case of a global alignment, but the traceback procedure must start at a position of a maximal value in \(t_S\) rather than at position \((m - 1, n - 1)\).

### 3 Aligning 2D Patterns

Let us now assume that we consider two rectangular patterns \(X = X[0 \ldots m_1 - 1, 0 \ldots n_1 - 1]\) and \(Y = Y[0 \ldots m_2 - 1, 0 \ldots n_2 - 1]\), of size \(M = m_1 \times n_1\) and \(N = m_2 \times n_2\) respectively. Each element \(X[i, j]\) with \(0 \leq i \leq m_1 - 1\) and \(0 \leq j \leq n_1 - 1\) and \(Y[k, \ell]\) with \(0 \leq k \leq m_2 - 1\) and \(0 \leq \ell \leq n_2 - 1\) belongs to the alphabet \(\Sigma\).

We want to compute the minimum cost of operations of insertion, deletion or substitution of individual symbols to transform \(X\) into \(Y\). Symbols can be inserted, deleted and substituted separately or in sub-rows or sub-columns.

#### 3.1 Global alignment

Aligning \(X\) and \(Y\) using dynamic programming consists in generalizing the recurrence formulas used for string alignment.

To that aim we need four two-dimensional tables \(D_R\), \(D_C\), \(I_R\) and \(I_C\) defined as follows for \(0 \leq i \leq m_1 - 1\), \(0 \leq j \leq n_1 - 1\), \(0 \leq i \leq m_2 - 1\) and \(0 \leq j \leq n_2 - 1\):

\[
\begin{align*}
D_R[i, j] &= \sum_{p=0}^{i} \text{Del}(X[i, p]), \\
D_C[i, j] &= \sum_{p=0}^{i} \text{Del}(X[p, j]), \\
I_R[i, j] &= \sum_{p=0}^{j} \text{Ins}(Y[i, p]), \\
I_C[i, j] &= \sum_{p=0}^{j} \text{Ins}(Y[p, j]).
\end{align*}
\]

In words, \(D_R[i, j]\) is the cost of the deletion of the prefix of length \(j + 1\) of row \(i\) of \(X\), \(D_C[i, j]\) is the cost of the deletion of the prefix of length \(i + 1\) of column \(j\) of \(X\), \(I_R[i, j]\) is the cost of the insertion of the prefix of length \(j + 1\) of row \(i\) of \(Y\) and \(I_C[i, j]\) is the cost of the insertion of the prefix of length \(i + 1\) of row \(j\) of \(Y\).

The tables \(D_R\) and \(D_C\) can be computed in time and space \(O(m_1 \times n_1)\). The tables \(I_R\) and \(I_C\) can be computed in time and space \(O(m_2 \times n_2)\).

We will also use two four-dimensional tables \(R\) and \(C\) of size \(m_1 \times n_1 \times m_2 \times n_2\) defined as follows:

\[
\begin{align*}
R[i, j, k, \ell] &= d(X[i, \ldots, j], Y[k, 0 \ldots \ell]) \\
C[i, j, k, \ell] &= d(X[0 \ldots i, j], Y[0 \ldots k, \ell]).
\end{align*}
\]

In words, \(R[i, j, k, \ell]\) contains the distance between the prefix of length \(j + 1\) of row \(i\) of \(X\) and the prefix of length \(\ell + 1\) of row \(k\) of \(Y\). Similarly, \(C[i, j, k, \ell]\) contains the distance between the prefix of length \(i + 1\) of column \(j\) of \(X\) and the prefix of length \(k + 1\) of column \(\ell\) of \(Y\).

The two tables \(R\) and \(C\) can be computed in time and space \(O(m_1 \times n_1 \times m_2 \times n_2)\).

Then, we use a four dimensional table \(T\) of size \((m_1 + 1) \times (n_1 + 1) \times (m_2 + 1) \times (n_2 + 1)\) defined as follows:

\[
T[i, j, k, \ell] = \min\{\text{cost of } \gamma | \gamma \in \Gamma_{X,Y}\}
\]

where \(\Gamma_{X,Y}\) is the set of all the sequences of edit operations that transform \(X\) into \(Y\), and the cost of an element \(\gamma \in \Gamma_{X,Y}\) is the sum of the costs of its elementary edit operations.

The values of the table \(T\) can be computed as follows for \(0 \leq i \leq m_1 - 1\), \(0 \leq j \leq n_1 - 1\), \(0 \leq k \leq m_2 - 1\) and \(0 \leq \ell \leq n_2 - 1\) (see Fig. 1):
3.2 Local alignment

To compute a local alignment between two rectangular two-dimensional patterns we will use two four-dimensional tables $R_S$ and $C_S$ of size $m_1 \times n_1 \times m_2 \times n_2$ defined as follows:

$$R_S[i, j, k, ℓ] = s(X[i, 0 . . j], Y[k, 0 . . ℓ])$$

and

$$(1) \quad C_S[i, j, k, ℓ] = s(X[0 . . i, j], Y[0 . . k, ℓ]).$$

Those two tables can be computed using the usual recurrence formulas for strings in time and space $O(m_1 \times n_1 \times m_2 \times n_2)$.

Then we use a four-dimensional table $T_S$ of size $(m_1 + 1) \times (n_1 + 1) \times (m_2 + 1) \times (n_2 + 1)$.

Let us denote: $r = R_S[i, j, k, ℓ]$, $c = C_S[i, j, k, ℓ]$, $r' = R_S[i - 1, j, k - 1, ℓ]$, $c' = C_S[i - 1, j, k - 1, ℓ - 1]$ and $q = Del_S(X[i, j]) + Ins_S(Y[ℓ, k])$.

The values of the table $T_S$ can be computed by the equation given below (see Fig. 2):

$$T_S[i, j, k, ℓ] = \max$$

$$\begin{cases}
T_S[i, j, k, ℓ] + Del_S(X[i, j]) \\
T_S[i - 1, j, k, ℓ] + Del_S(X[i, j]) \\
T_S[i, j - 1, k, ℓ] + Del_S(X[i, j]) \\
T_S[i, j, k - 1, ℓ] + Del_S(Y[k, ℓ]) \\
T_S[i, j, k, ℓ - 1] + Ins_S(Y[ℓ, k]) \\
T_S[i - 1, j, k - 1, ℓ] + Ins_S(X[i, j])
\end{cases}$$

(2)

for $0 \leq i \leq m_1 - 1$, $0 \leq j \leq n_1 - 1$, $0 \leq k \leq m_2 - 1$ and $0 \leq ℓ \leq n_2 - 1$.

We have the following margin initializations:

$$T_S[-1, j, k, ℓ] = (k + 1) \times (ℓ + 1)$$

and

$$T_S[i, j - 1, k, ℓ] = (i + 1) \times (j + 1)$$

for $0 \leq i \leq m_1 - 1$, $0 \leq j \leq n_1 - 1$, $0 \leq k \leq m_2 - 1$ and $0 \leq ℓ \leq n_2 - 1$.

Then a traceback can be performed from $T_S[1, 1, 1, 1]$ as in the two sequence comparison case.

Fig. 1. (a): deletion of $X[i, 0 . . j]$; (b): deletion of $X[0 . . i, j]$; (c): insertion of $Y[k, 0 . . ℓ]$; (d): insertion of $Y[0 . . k, ℓ]$; (e): substitution of $X[i, 0 . . ℓ]$ by $Y[k, 0 . . ℓ]$; (f): substitution of $X[i, 0 . . i, j]$ by $Y[0 . . k, ℓ]$; (g): substitution of $X[i, 0 . . i, j]$ by $Y[k, 0 . . ℓ]$ and substitution of $X[0 . . i - 1, j]$ by $Y[0 . . k - 1, ℓ]$; (h): substitution of $X[0 . . i - 1, j]$ by $Y[0 . . k - 1, ℓ]$ and substitution of $X[i, 0 . . ℓ]$ by $Y[0 . . k, ℓ]$.

3.2.1 Non-rectangular patterns

When considering non-rectangular patterns, it is enough to embed the patterns in the smallest rectangles that can contain them and fill up the empty positions with a don’t care symbol $*$ such that $d(\ast, \ast) > 0$ and $d(\ast, \ast) = 0$ when considering distances and $s(\ast, a) = s(a, \ast) < 0$ and $s(\ast, \ast) > 0$ when considering similarity scores.

3.2.2 Unit scores

When considering unit scores, the tables $D_R, D_C, I_R$ and $I_C$ are useless since their values can be computed in constant time.
and the four best local alignments can then be merged.

Sub 1

\begin{align*}
\text{Maximal value in} \quad 1) & \leq 0 \\
\text{alignment of} \quad 0 & 0 \\
\text{with} \quad x & x \\
\text{alignment of} \quad 0 & 0 \\
\text{with} \quad x & x \\
\text{insertion of} \quad Y & Y \\
\text{and alignment of} \quad 0 & 0 \\
\text{with} \quad X & X \\
\text{insertion of} \quad X & X \\
\text{insertion of} \quad X & X \\
\text{with} \quad Y & Y \\
\text{alignment of} \quad 0 & 0 \\
\text{and alignment of} \quad 0 & 0 \\
\text{with} \quad x & x \\)

This method can be applied in the four orientations and the four best local alignments can then be merged. All this can still be performed with the same complexi-

Fig. 2. (a): deletion of \(X[i,j]\) and alignment of \(x[0...i-1,0...j]\) with \(Y[0...k,0...\ell]\); (b): deletion of \(X[i,j]\) and alignment of \(x[0...i,0...j-1]\) with \(Y[0...k,0...\ell]\); (c): insertion of \(Y[k,\ell]\) and alignment of \(x[0...i,0...j]\) with \(Y[0...k-1,0...\ell]\); (d): insertion of \(Y[0,\ell]\) and alignment of \(x[0...i,0...j]\) with \(Y[0...k,0...\ell-1]\); (e): substitution of \(X[i,j]\) by \(Y[k,\ell]\) and alignment of \(x[0...i-1,0...j]\) with \(Y[0...k-1,0...\ell]\); (f): substitution of \(X[i,j]\) by \(Y[k',\ell]\) and alignment of \(x[0...i-1,0...j-1]\) with \(Y[0...k,0...\ell-1]\); (g): substitution of \(X[i,j']\) by \(Y[k,\ell']\) and substitution of \(X[i',j]\) by \(Y[k',\ell]\) and alignment of \(x[0...i-1,0...j-1]\) with \(Y[0...k-1,0...\ell-1]\); (h): substitution of \(X[i,j']\) by \(Y[k,\ell']\) and substitution of \(X[i',j]\) by \(Y[k',\ell]\) and alignment of \(x[0...i-1,0...j-1]\) with \(Y[0...k-1,0...\ell-1]\). (i): substitution of \(X[i,j]\) by \(Y[k,\ell]\) and alignment of \(x[0...i-1,0...j]\) with \(Y[0...k-1,0...\ell]\); (j): substitution of \(X[i,j]\) by \(Y[k,\ell]\) and alignment of \(x[0...i,0...j-1]\) with \(Y[0...k,0...\ell-1]\); (k): substitution of \(X[i,j]\) by \(Y[k,\ell]\) and alignment of \(x[0...i,0...j-1]\) with \(Y[0...k-1,0...\ell-1]\); (l): substitution of \(X[i,j]\) by \(Y[k,\ell]\) and alignment of \(x[0...i-1,0...j]\) with \(Y[0...k-1,0...\ell]\); with \(0 \leq \ell' \leq \ell\).

\hspace{1cm}

and

\[ T_S[i,j,-1,\ell] = T_S[i,j,k,-1] = 0 \]

for \(0 \leq i \leq m_1-1, 0 \leq j \leq n_1-1, 0 \leq k \leq m_2-1\) and \(0 \leq \ell \leq n_2-1\).

The values of the table \(T_S\) of size \((m_1+1) \times (n_1+1) \times (m_2+1) \times (n_2+1)\) for detecting similar rectangular sub-patterns can be performed in time and space \(O(m_1 \times n_1 \times m_2 \times n_2)\).

The traceback procedure must start at a position of a maximal value in \(T_S\) rather than at position \((m_1-1,n_1-1,m_2-1,n_2-1)\).

Example: With the following score system: \(Sub_S(a,a) = 1, Sub_S(a,b) = -1\) and \(Ins_S(a) = Del_S(a) = -1\), for \(a, b \in \Sigma\) such that \(a \neq b\), Fig. 3 shows on the left the two patterns to align and on the right the best local alignment where symbol \(c\) on the third line of the second pattern is inserted, while symbols \(ghi\) of the third line of the first pattern are substituted by symbols \(aba\) of the fourth line of the second pattern.

This method can be applied in the four orientations and the four best local alignments can then be merged. All this can still be performed with the same complexi-

Fig. 3. Local alignment of 2D patterns.

4 APPLICATIONS

The study of log files and annotated corpus dialogues aims most of the time identifying significant recurrent behaviors. These behaviors appear as repeated patterns, which can be represented thanks to automata [26], timed automata [19], Petri networks [17], diagram sequences [24], and so on. All these representations are based on linear patterns.

In the particular case of corpus of dialogues whose annotations characterize an emotional process, each location must be annotated by a series of terms. It necessitates a non-linear research of patterns (i.e. two-dimensional patterns), therefore conventional approaches cannot be used. The approach presented here, using 2D alignments grounds, responds to this problem. It is a three-step process:

1) identification of 2D patterns through the algorithm presented previously;
2) assessing the semantic relevance of the detected patterns (expertise);
3) statistical evaluation of those patterns.

The algorithm was tested on dialogues between parents and four year old children during the narration of two child stories (A and B). Each statement has been transcribed and coded according to a grid ([7]) and corresponds to one line. This grid was used as part of a psychological study on mentalist interpretations of the behavior of a character. The aim is to identify dialogical, semantical and pragmatical characteristics of the discourse that adults use to explain the behavior of a character who is confronted with a false belief [25]. This grid is composed of five columns, each column includes between 2 and 7 encoding terms (6, 3, 7, 2, 4): the terms of the first column are related to the nature of the utterance. It can be an affirmation (a or A), a query (q or Q), a request to pay attention to the story (D), or a demand for general attention (G). The second column concerns the statement referring. The statement may refer to the character (P), the interlocutor (H) or the speaker (R). The third column describes the mental states. The dialogue partners can describe the emotion (E), volition (V), observable or not observable cognition (B or N), the epistemic (K), the assumption...
(Y) and the surprise (S). The surprise is distinguished from the emotion because of the link with the belief from which it is issued. The last two columns are devoted to explanations by cause / consequence (C), opposition (O) or empathy (M), which can be applied either to explain the story (J), or to explain a situation with a personal context (F). For each of the five columns a symbol is used to represent the absence of information: (, [, ], ), ] respectively.

For instance, line 35 of dialogue of Fig. 5 “but Babar doesn’t know that it’s in” is coded in the following way: this statement contains a mental state, thus A in the first column; this mental state refers to the character (“Babar”), thus P in the second column; this mental state (“know”) refers to non observable cognition, thus N in the third column; “but” denotes a justification by opposition, thus O in the fourth column; the context refers to the story itself, thus J in the fifth column.

A substitution matrix was specifically designed for this application (see Figure 4). Each pair of coding values is given a score, the closer the coding values the higher the score. The scores go from 0 to 10. Actually the designed scoring system gives high scores to pair of coding values among one category (mental state, referring, ...) And inside these categories, there exist also some variations. For instance the score of code “a” and code “C” is 0, indeed code “a” means that the statement is an affirmation which is a grammatical information while code “C” denotes that the statement is a justification by consequence which is an information on the statement content. These two codes are not at the same level of information thus a score of 0 so that the dynamic programming method will clearly distinguish them. On the other hand, pair of codes in a same category are given higher scores. Pairs in the set of codes \{E, V, B, N, K, Y, S\} that forms the mental state category are given a score greater or equal than 5. Emotion (E) and volition (V) are theoretically very close since emotion is dependant from desire (the child wants a toy, and if he gets it he will be happy otherwise he will be sad). Thus the score for these two codes is high (9).

For this study we focused on mental states and their justifications. Thus we put aside grammatical information. Indeed we gave higher scores to pairs of codes inside those categories: referring (second column), mental states (third column), justifications (fourth column). We deliberately distinguished the codes of the first column from the others, and this is why the pattern extracted by our method do not take into account codes in this first column (see below).

After aligning all pairs of different dialogues and intra-align dialogues, we focused on three dialogues. Two dialogues concern story A, and the third one concerns story B; the dialogue partners are all different. This technique enabled us to identify fuzzy patterns repeated inside one dialogue and within the other two dialogues. On series of 12, 14 and 16 lines (depending on the location) a pattern, divided into two parts, appears: the first set is composed of a series of lines (9, 11 and 14) which consists of terms related with the presence of a mental state and a referring. This first set is followed by a series of lines (2 or 3) that contains a justification associated with a mental state.

Figures 5 to 7 describe the patterns discovered by the method (gray parts), one in story A and one in story B. To illustrate our point, mental states and the statement referring contain in the fuzzy patterns are in capital letters in the dialogue and correspond to the second and the third column. In the same way, the explanation and its context are in bold face and upper case letters in the dialogue and correspond to the two last column.

The interest of this pattern is that it seems to characterize both dialogic organization and content, difficult or even impossible to observe to the naked eye. For this example, it appears that parents ensure to prepare the “mentalist” justification of the situation. Indeed, as a first step, they seem to describe and explain the mental states involved, then in a second time, they clarify the cause, consequence or opposition that explain the mental states of a character and the character’s behavior.

Eventually, all identified patterns will be listed and searched for in all dialogues (we are looking for fuzzy occurrences). Thus, for every terms of all dialogues, it will be possible to know to which patterns it belongs and what are the dialogues (with the positions) in which these patterns occur. The aim is to assist an expert to detect significant patterns.

5 Conclusion and Perspectives
We presented a very generic method to align two dimensional patterns using dynamic programming in quadratic time and space. This method enables the computation of global and local alignments. A first challenge consists in visualizing the computed alignments. Many questions arises. Is it possible to align 2D patterns in sub-quadratic time as for sequences? (see [9]). Is it possible to develop a heuristic similar to BLAST ([1]) to quickly align a 2D pattern against a data bank of 2D patterns? On the other hand, it is necessary to accompany the proposed method of a validation to estimate the statistical relevance of the results. Beyond the scientific and theoretical interest, this method may eventually not only provide a tool for diagnosis in the context of asymmetric dialogical situations but more generally could provide a tool for interview training. Indeed, during an interaction, emotions that are not explicitly expressed appeals to the subjectivity of the interlocutor to be interpreted. These emotions, necessary for an efficient dialogue, are especially crucial during diagnosis interactions. Imagine a young child who has consulted for an abdominal pain. The child and the physician do not have neither the same level of language, nor the same representation of body and pain level. Yet, the prescription of pain medication can be achieved only on the basis of what the child says about his pain. This method, completed with a semi-directive interview could enable to catch information not
Table 1: Substitution matrix.

| a | q | A | Q | G | D | P | R | H | C | O | M | J | F | E | B | N | V | K | Y | S |
| a | 10 | q | 9 | 10 | A | 10 | 9 | 10 |
| G | 2 | 8 | 2 | 8 | 10 | D | 2 | 8 | 2 | 8 | 10 | 10 | P | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 10 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 9 | 10 |
| H | 0 | 0 | 0 | 0 | 0 | 0 | 7 | 8 | 10 |
| C | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 2 | 2 | 10 |
| O | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 2 | 2 | 9 | 10 |
| M | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 8 | 2 | 8 | 8 | 10 |
| J | 0 | 0 | 0 | 0 | 0 | 0 | 7 | 4 | 3 | 1 | 1 | 2 | 10 |
| F | 0 | 0 | 0 | 0 | 0 | 0 | 4 | 7 | 3 | 1 | 1 | 7 | 9 | 10 |
| E | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 7 | 5 | 3 | 3 | 8 | 2 | 4 | 10 |
| B | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 5 | 5 | 3 | 3 | 3 | 4 | 4 | 8 | 10 |
| N | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 5 | 5 | 3 | 3 | 3 | 4 | 4 | 8 | 10 |
| V | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 5 | 5 | 3 | 3 | 3 | 2 | 4 | 9 | 7 | 7 | 10 |
| K | 0 | 0 | 0 | 0 | 0 | 0 | 7 | 5 | 5 | 3 | 6 | 3 | 4 | 4 | 8 | 7 | 8 | 7 | 10 |
| Y | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 5 | 5 | 4 | 3 | 3 | 4 | 4 | 8 | 7 | 8 | 7 | 9 | 10 |
| S | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 5 | 5 | 3 | 3 | 6 | 3 | 4 | 9 | 7 | 7 | 7 | 9 | 8 | 10 |

Fig. 4. Substitution matrix.

P | don’t worry | A | P | E |
---|---|---|---|---|
25 | P | THEY ARE HIDING themselves | A | P | B |
26 | P | they are looking for | a | f |
27 | P | who could have taken away the crown | q | f |
28 | C | it’s in ! The crown is inside the box ! | a | f |
29 | P | so THEY are SUSPECTING a lot of people Cornelius, Celeste and the old lady | A | P | Y | C | J |
30 | P | who could have stolen the crown | q | f |
31 | C | the crown it’s in ! | a | f |
32 | P | do YOU BELIEVE? | Q | H | K |
33 | C | yes | a | f |
34 | P | BUT BABAR DOESN’T KNOW that it’s in | A | P | N | O | J |
35 | P | SO HE TELLS HIMSELF that is a bomb, the crown | A | P | N | C | J |
36 | P | or I don’t KNOW what | A | R | N |
37 | C | the crown | a | f |
38 | | | | |

Fig. 5. Story A.

P | where is it? | q | f |
---|---|---|---|
9 | P | BABAR OBSERVES Cornelius give a pack to a stranger | A | P | B |
10 | P | what’s inside? | q | f |
11 | P | who is this masked stranger? | q | f |
12 | P | who has stolen the crown? | q | f |
13 | P | BABAR UNCOVERS the masked stranger! | Ä | P | B |
14 | P | it’s the queen Celeste! | a | f |
15 | P | HE is ASKING himself questions | A | P | N |
16 | P | why the queen Celeste disguise herself? | q | f |
17 | P | Babar goes and sees the old lady to ask her about it | a | f |
18 | C | yes | a | f |
19 | P | THE OLD LADY doesn’t WANT him to go inside! | A | P | V |
20 | P | BUT just behind HER was HIDDEN | A | P | B | O | J |
21 | P | a SURPRISE for him IN FACT | A | P | S | O | J |
22 | P | and then Babar goes back home | a | f |
23 | P | and every body was here with a big gift-wrap | a | f |
24 | | | | |

Fig. 6. Story A.
and suddenly oups everybody fall down in the water

Do you believe

Are you sure?

The donkey says

and catch it

Everybody watch him

Suddenly the swan mummy comes back

It’s a boat?

What’s happening?

They sing “bateau sur l’eau” [French nursery rhyme]

And the donkey grab it in its mouth

They go on board

The little swan wants to go on board too

Because it’s a boat

The ducks says “nooo!”

But it doesn’t matter

Everybody can swim

The duck can swim

Suddenly the swan mummy comes back

Look it’s a boat

Easily perceptible necessary to diagnose. In addition, this method joint with a psychological expertise raise techniques and effective patterns (dialogical or not). Thus, training to the interview techniques could be improved. We think that this method could help in real time a trainee during an interview training session.

Fig. 7. Story B.

REFERENCES


Artificial Intelligence and the 18th Innovative Applications of Artificial Intelligence Conference, Boston, Massachusetts, USA, 2006.

